Accountable Transaction Inclusion Lists: Enhancing Ethereum's Censorship Resistance

Patrick Spiesberger
Karlsruhe Institute of Technology, Germany

Hannes Hartenstein
Karlsruhe Institute of Technology, Germany

Abstract

In Ethereum, transaction inclusion is rarely in question; what matters is the delay until inclusion. Currently, block builders could exercise censorship across consecutive blocks, threatening time-critical applications, such as on-chain auctions. To mitigate this risk, existing proposals such as FOCIL, scheduled for deployment in late 2026, assign a committee to list transactions for mandatory inclusion. However, no committee member is held accountable for the actual inclusion of the transactions: an adversary can bribe the entire committee to omit any transaction for less than 2 € per block under current conditions. We argue that accountability, i.e., requiring all exclusion decisions to be publicly disclosed and verifiably complete, with violations attributable to a specific party, substantially raises censorship costs. To this end, we propose Fair Forward Inclusion Lists (FairFIL) as an accountable censorship resistance mechanism for Ethereum. In FairFIL, every builder must publish all transactions the builder chooses to censor, subject to a protocol-anchored policy; a committee verifies the completeness and validity of this disclosure. The subsequent builder must include these transactions, forfeiting the full block reward upon any omission. Therefore, under FairFIL, extending censorship beyond a single slot requires an assembler to forfeit a full block reward. We show that compliance is rational for all participants within our behavior model. Our empirical evaluation on Ethereum mainnet indicates that multi-block censorship costs one order of magnitude more than under existing proposals, while leaving the builder's MEV extraction freedom largely intact. Initial measurements further suggest that the mempool consistency FairFIL requires is met in practice.

2012 ACM Subject Classification Computer systems organization → Reliability

Keywords and phrases Ethereum, Blockchain, Censorship Resistance, Inclusion Lists, Fairness

Acknowledgements This work was funded by the KASTEL Security Research Labs.

1 Introduction

Ethereum [6], as of mid 2026, is the largest general-purpose smart-contract platform by market capitalization, hosting a wide range of applications that depend on transaction inclusion with low and predictable latency. Ethereum operates on a time-slotted model [12]: in each 12 s slot, a single leader (in Ethereum terminology the *proposer*) is selected at random to assemble a block, while the remaining participants (*validators*) attest to its validity. During the slot, the proposer has short-lived but unilateral control over which transactions are included in the block and in what order [20, 29]. Correspondingly, any transaction can be excluded at will by the proposer. For some transactions, inclusion within a handful of slots is sufficient. However, a broad class of transactions is time-critical: each additional slot of inclusion latency can diminish the value delivered to the user or application, or more broadly undermine Ethereum's attractiveness as a reliable platform. Examples include auctions, decentralized finance, and commit-and-reveal protocols. In auctions, a bid submitted shortly before the auction closes may be suppressed after the deadline [18, 39]. In decentralized finance, a stop-loss transaction submitted when a trigger price is reached can be censored; each additional delay allows the asset price to move further against the user's intent [9, 26].

arXiv:2607.22040v1 [cs.DC] 24 Jul 2026

In commit-and-reveal schemes, a reveal transaction must be submitted within a bounded window after committing; if censored, the protocol cannot distinguish the outcome from an intentional non-reveal, and the participant may be penalized despite having acted in time.

For such applications, the essential question is not whether a transaction will eventually be included in the blockchain, but how quickly the transaction affects the application. We follow the established usage in the Ethereum literature [7, 37] and use the term "censorship" for additional inclusion latency – the exclusion of a transaction from one or more consecutive blocks – rather than for permanent exclusion from the blockchain [18]. As long as control over inclusion decisions rotates frequently among independent parties, no single party can delay a transaction beyond a limited timeframe. Concerns arise when one party controls block construction over extended periods, a scenario facilitated by Ethereum's Proposer–Builder Separation (PBS) [22, 10]. Under PBS, proposers delegate block construction to specialized builders to capture Maximal Extractable Value (MEV), i.e., the additional revenue obtained by strategically including, excluding, or reordering transactions within or across blocks [20]. Empirical evidence shows that two builders assembled more than 80 % of all blocks in 2024 and 2025 [40, 2], and nearly all proposers rely on them [32]. A dominant builder could, in principle, impose censorship over many consecutive slots, thereby undermining the reliability of the network. Our measurements, as described below, underline the relevance in practice: in September 2025, an average of 8.3 transactions per block were censored despite being eligible for inclusion under the intended tip-based reference ordering [8].

To mitigate transaction censorship, prior work proposes mechanisms that distribute inclusion control across a committee of κ participants per slot [27, 34, 36, 19]. The common idea is to split one party's exclusive control into shares across κ parties, such that a single honest member suffices to force the inclusion of a transaction. However, as we show later, an adversary can bribe the entire committee of any of these mechanisms at a cost of only a few euros per slot under current conditions. We attribute this low cost of censorship to the fact that no individual committee member is held accountable: no member's exclusions are publicly attributable, verifiable, or subject to protocol-enforced penalties. In this paper, we therefore address the following research question:

Research question. *Does accountability for exclusion decisions substantially increase the cost of censoring a transaction?*

Contribution and Rationale. The idea of accountability in censorship-resistance mechanisms and an initial sketch of FairFIL were first introduced in [33]. This paper builds on that foundation and makes three main contributions. First, we formalize accountability for censorship resistance in Ethereum. We prove that accountability can substantially increase the cost of sustained censorship: under accountable inclusion enforcement, a censoring builder forfeits the entire block reward, increasing censorship costs by roughly one order of magnitude compared to prior approaches. Second, we present Fair Forward Inclusion Lists (FairFIL), an accountable censorship resistance mechanism, and prove that under a fully rational participant model, FairFIL enforces block invalidation under sustained censorship. Third, we implement FairFIL and empirically evaluate the mechanism on Ethereum mainnet, quantify the cost of censorship and provide first evidence for practical deployability. FairFIL is designed to be explicitly MEV-friendly, preserving most of the builders' flexibility for MEV extraction. Two considerations motivate this choice of an "MEV-friendly design". First, MEV constitutes a substantial share of proposer revenue [21]. Curtailing MEV-extraction freedom risks undermining Ethereum's long-term viability. Though difficult to quantify, we regard this risk as significant, as does prior work [5]. Second, over 90 % of validators [22, 32]

voluntarily adopted MEV-Boost [15], despite censorship and centralization risks [22, 40], demonstrating that validators will deviate from protocol intentions when compliance reduces revenue. FairFIL trades only a small share of the block assembler's MEV-extraction freedom in exchange for inclusion guarantees: any transaction may be censored for one slot but must be included thereafter, keeping assembler constraints low while guaranteeing timely inclusion for the transaction sender.

Paper structure. Section 2 introduces the system model, defines transaction censorship, and states the participant and adversary model. Section 3 surveys prior censorship resistance mechanisms and identifies the absence of accountability as their main limitation. Section 4 presents the FairFIL protocol. Section 5 empirically evaluates FairFIL on Ethereum mainnet, formalizes accountability as four properties satisfied by FairFIL, and quantifies the per-slot cost of sustained censorship compared to prior mechanisms. Section 6 discusses assumptions and deployment considerations regarding MEV; Section 7 concludes.

2 Foundations and Design Space

2.1 System Model and Transaction Censorship

We consider an abstracted Ethereum model in which the blockchain is extended in discrete time slots. For each slot s, a committee $V_s \subset V$ of validators is pseudo-randomly selected from the active validator set V, while a single proposer $v_s^P \in V_s$ is responsible for assembling the block at the beginning of slot s [12]. We refer to the party that actually constructs the block – whether the proposer itself or a delegated builder under PBS [10] – as the *block assembler* β_s. Table 3 summarizes the symbols and abbreviations used throughout this paper.

Transactions. A transaction τ is a signed user statement [12] specifying an intent and an execution cost $\text{gas}(\tau)$ determined upon execution. For every unit of gas consumed, the user pays a voluntary per-gas tip $f^{\tau}_{\text{tip, gas}}$ – intended to incentivize faster inclusion – to the block assembler β_s, and a mandatory per-gas base fee $f^{\tau}_{\text{base, gas}}$ that is received by no party [8]. Accordingly, the total per-transaction tip paid to block assembler β_s for including τ is $f^{\tau}_{\text{tip, tx}} := f^{\tau}_{\text{tip, gas}} \cdot \text{gas}(\tau)$, and the total per-transaction base fee is $f^{\tau}_{\text{base, tx}} := f^{\tau}_{\text{base, gas}} \cdot \text{gas}(\tau)$.

Mempool. Transactions are disseminated through Ethereum's peer-to-peer (P2P) layer and stored, at each node, in a local buffer called the mempool. We write Pool_s^n for the mempool view of a node n at slot s; in particular, $\text{Pool}_s^{\beta_s}$ is the block assembler's view. For the theoretical analysis, we assume all validators share an identical mempool view, consistent with prior analyses of censorship resistance mechanism designs [36]. In practice, mempools are not perfectly consistent; we revisit this assumption empirically in Section 5.2 and find that the near-perfect consistency observed appears to be sufficient for FairFIL operation.

Block construction and validity. At the beginning of slot s, the block assembler selects a set of transactions from $\text{Pool}_s^{\beta_s}$, possibly augmented with transactions received via a private communication channel (Exclusive Order Flow, XOF) [20]. Subsequently, the assembler orders and executes these transactions and publishes the resulting block B_s. A block is protocol-valid if the execution on top of the state induced by block B_{s-1} respects Ethereum's execution rules [38]; in particular, the cumulative gas consumed by B_s must not exceed the per-slot gas limit $g_{\max}(B_s)$. A block is accepted once a majority of V_s attests to its validity; due to Ethereum's proposer boost, slightly less than a majority suffices in practice [30].

Transaction Censorship. In this work, we restrict the notion of censorship to a specific operational instance: the exclusion of a transaction from the block currently being assembled. Censorship in this sense denotes additional inclusion latency, not permanent exclusion from the blockchain. We neither consider self-censorship (users refraining from submitting transactions) nor censorship at Ethereum's P2P layer. To decide which transactions should have been included, we adopt a deterministic reference ordering. Transactions in the mempool $\text{Pool}_s^{\beta_s}$ are sorted in descending order of their per-gas tip $f^{\tau}_{\text{tip, gas}}$, with the transaction hash as a tie-breaker. Iterating through this sequence, we admit every transaction whose execution preserves block validity; we call the resulting sequence the *Fair Ordering*[1] of the mempool.

▶ **Definition 1** (Transaction censorship). *A transaction τ is* censored *from block B_s if τ was propagated before the start of slot s via Ethereum's P2P layer and its inclusion in B_s under the Fair Ordering would preserve block validity, yet transaction τ is excluded from block B_s.*

The condition "propagated before the start of slot s" is not precisely observable in a decentralized system. In practice, we operationalize this condition as the initial propagation at least 3 s before the start of the slot to account for network propagation delays. Table 1 illustrates Definition 1 on a mempool $\text{Pool}_s^{\beta_s} = \{\tau_1, \ldots, \tau_5\}$, augmented with a privately submitted transaction τ_{private}, against a block gas limit of $g_{\max} = 100$ gas. The candidate sequence obtained by sorting Pool_s by per-gas tip in descending order is $[\tau_1, \tau_2, \tau_3, \tau_4, \tau_5]$. Iterating through this sequence and admitting each transaction whose execution still fits into $g_{\max}$ yields the Fair Ordering $[\tau_1, \tau_2, \tau_4, \tau_5]$ with cumulative gas 50. Transaction τ_3 is dropped because including $\{\tau_1, \tau_2, \tau_3\}$ would exceed $g_{\max}$. By Definition 1, any transaction in the *Fair Ordering* not included in B_s is censored. In our example, this applies to τ_2, as well as to τ_4, which is displaced by the privately submitted τ_{private}.

τ	Tip $f^{\tau}_{\text{tip, gas}}$	gas(τ)	Includable?	$\in B_s$?	Censored?
τ_5	2	10	Yes	✓	No
τ_2	8	10	Yes	×	**Yes**
τ_3	7	90	No	×	No
τ_{private}	20	70	Yes	✓	–
τ_4	5	10	Yes	×	**Yes**
τ_1	10	20	Yes	✓	No

Table 1 Set of transactions available to the current block assembler for block construction ($g_{\max} = 100$), resulting in block $B_s = [\tau_5, \tau_{\text{private}}, \tau_1]$. While this selection may appear arbitrary, it reflects the assembler's MEV extraction strategy, which encompasses both the censorship of transactions and the inclusion of private transactions, with the objective of maximizing revenue.

We further classify censored transactions by how long their inclusion is delayed.

▶ **Definition 2** (n-slot censorship). *Let s be the earliest slot in which transaction τ is includable. For $n \in \mathbb{N}_+$, τ experiences n-slot* censorship *if it is censored from blocks $B_s, \ldots, B_{s+n-1}$ and included in block B_{s+n}. The case $n = 0$ corresponds to the absence of censorship for τ.*

2.2 Behavior Model and Censorship Resistance Mechanisms

We apply the Byzantine, Altruistic, Rational (BAR) model by Aiyer et al. [1] to characterize validator and block assembler behavior, following prior work such as [32]. An *altruistic*

[1] This use of "fair" differs from receive-time-based notions of order fairness such as the Aequitas family [24].

participant follows the protocol's intent regardless of payoff. We assume their absence, as protocols should be robust even in the worst plausible case, and revisit this assumption in Section 6.2. In what follows, all validators and block assemblers are modeled as *rational participants*: a rational participant deviates from the protocol whenever deviation yields a strictly higher personal payoff than compliance, and may cooperate with other rational participants. Such a deviation need not violate the protocol; it may instead exploit degrees of freedom the protocol permits, contrary to its intent. We treat censorship as a deviation from the Ethereum protocol [25] and later extend this to deviations from censorship-resistance protocols. When compliance and deviation yield identical payoffs, we assume compliance: deviating offers no financial gain, while 'unfair' behavior could undermine Ethereum's long-term viability and thereby reduce future validator revenue. However, we assume that any strictly positive revenue gain suffices to induce deviation. Adversary $\mathcal{A}$ is not profit-driven and aims to censor transactions: $\mathcal{A}$ can communicate directly with all participants [36] and may bribe rational participants to deviate, subject to a per-slot bribing budget $\mathcal{M}$. All bribes are paid on-chain via a bribing smart contract [31]; out-of-band payments (e.g., bank transfers) are outside our model. Each bribe therefore incurs a fixed overhead c_{bribe} for adversary $\mathcal{A}$, covering fund transfer and on-chain verification – in addition to the bribe amount itself.

Budget restriction. In Ethereum, censorship resistance is fundamentally an economic property [18]. As long as block assemblers may choose to withhold a block, any sufficiently wealthy adversary can censor transactions by compensating the block assembler β_s for the forgone block reward $\mathcal{R}(B_s)$, defined as the sum of transaction tips and extracted MEV. Providing censorship resistance against arbitrarily wealthy adversaries would require substantial structural changes to Ethereum and lies outside the scope of this paper. Accordingly, we assume that an adversary cannot bribe a block assembler to an extent that prevents the publication of a block. Under this assumption, the adversary's per-slot budget is bounded by the block reward $\mathcal{R}(B_s)$, since any larger budget would enable the adversary to prevent block publication in every slot, thereby violating Ethereum's liveness property and halting chain progression. We further assume that the adversary cannot bribe the slot committee V_s to accept an invalid block or reject a protocol-valid one. Such attacks would require bribing a majority of the committee, which comprises approximately 30 000 validators in 2025 [23]. We argue that the required budget significantly exceeds the block reward $\mathcal{R}(B_s)$.

Based on this adversary model, we define censorship resistance as follows:

▶ **Definition 3** ($(\mathcal{M},\sigma)$-censorship resistance)**.** *A protocol is* $(\mathcal{M},\sigma)$-censorship-resistant *if, for any transaction* τ*, an adversary with per-slot budget* $\mathcal{M}$ *cannot prevent* τ *from being included for more than* σ *consecutive slots under rational behavior of all other participants.*

We write $\mathcal{C}_\tau = [\mathcal{C}_\tau^s, \mathcal{C}_\tau^{s+1}, \dots]$ for the per-slot cost sequence incurred by $\mathcal{A}$ to censor τ, where each entry $\mathcal{C}_\tau^{s+i}$ denotes the cost in the respective slot $s+i$. The summation of costs up to the n-th element therefore corresponds to the cost of n-slot censorship of τ.

Ethereum baseline. We establish the censorship resistance of plain Ethereum as of early 2026. Informally, outbidding transaction τ's per-transaction tip $f^\tau_{\text{tip, tx}}$ by any $\epsilon > 0$ makes accepting the bribe more profitable than including τ, thereby censoring τ for one slot; the attack can be repeated in each slot. This yields Lemma 4; the proof is given in Appendix C.

▶ **Lemma 4** (Ethereum baseline). *Ethereum is* $\left(f^{\tau}_{\text{tip, tx}} + c_{\text{bribe}},\ 0\right)$*-censorship-resistant. The per-slot cost of excluding transaction* τ *is* $\mathcal{C}^{\text{Eth}}_{\tau} := f^{\tau}_{\text{tip, tx}} + c_{\text{bribe}} + \epsilon$ *for each slot, with* $\epsilon > 0$.

In absolute terms, a budget of 0.10 € per slot is sufficient to censor the median transaction, where transactions are ranked by total tips, observed between September and December 2025. The required budget is predominantly driven by the bribe overhead c_{bribe}.

While the per-slot cost of censorship is financially bounded by the block reward, the objective of a censorship resistance mechanism is to maximize the cost an adversary must incur to censor a target transaction. We consider inclusion list mechanisms that achieve this by designating a set of required transactions $\mathcal{T}_{\text{req}}$ whose inclusion is protocol-enforced. Accordingly, we regard the transaction set $\mathcal{T}_{\text{req}}$ as an obligation imposed on a specified party.

▶ **Definition 5** (Inclusion List-Based Censorship Resistance Mechanism, IL-CR). *An inclusion list-based censorship resistance mechanism* Π_n *designates, for each slot* s*, a set* $\mathcal{T}_{\text{req}}$ *of transactions and requires that every transaction in* $\mathcal{T}_{\text{req}}$ *is included no later than block* B_{s+n}*. Such a mechanism is called accountable if the construction of* $\mathcal{T}_{\text{req}}$ *and its enforcement are publicly verifiable, so that any deviation can be detected and penalized.*

We distinguish two classes of IL-CR mechanisms: a *constructing* mechanism assigns construction of $\mathcal{T}_{\text{req}}$ to a committee without content constraints, and is therefore non-accountable. In contrast, a *verifying* mechanism defines a protocol-anchored construction policy against which a committee verifies compliance – any deviation is detectable, regardless of who performed the construction. A verifying mechanism is *accountable* if every policy violation can be attributed to a specific party and the protocol specifies a corresponding consequence.

The significance of this distinction follows from the rationality model. In constructing mechanisms, each committee member independently contributes a partial list – a subset of transactions selected at the member's discretion – from which $\mathcal{T}_{\text{req}}$ is assembled as the aggregate. No member faces any penalty for omitting transaction τ from their partial list; any bribe exceeding the per-transaction tip therefore suffices to induce omission. Further, without a policy constraint, members may insert privately negotiated transactions, which can displace legitimate mempool transactions, enable advanced MEV strategies, and undermine external verifiability. Under the rationality assumption, this is the predicted outcome in practice: an outsourcing of $\mathcal{T}_{\text{req}}$ construction to specialized builders, analogous to PBS, thus becomes plausible. In verifying mechanisms with accountability, by contrast, any deviation exposes the constructing party to a financial penalty independent of transaction fees – such as forfeiting the full block reward $\mathcal{R}(B_s)$ – making omission far more costly than any per-transaction tip. We demonstrate in this paper that FairFIL, as an accountable verifying mechanism, raises the per-slot cost of sustained censorship by roughly one order of magnitude relative to the constructing mechanisms surveyed in the following Section 3.

3 Related Work

We consider four prior censorship resistance mechanism proposals: FIL [27], FOCIL [34], MCP [19], and AUCIL [36], where FIL, FOCIL, and AUCIL are inclusion list-based censorship resistance mechanisms. We further classify MCP as an IL-CR mechanism despite structural differences, which we discuss in detail below. The FIL approach represents the most basic instance of an IL-CR mechanism, where $\mathcal{T}_{\text{req}}$ is constructed by a single party ($\kappa = 1$). In contrast, the other mechanisms distribute the construction of $\mathcal{T}_{\text{req}}$ across a committee of size $\kappa > 1$ per slot. For each mechanism, we describe the design and derive the resulting

per-slot censorship cost, as detailed in Appendix F. To the best of our knowledge, no existing mechanism provides accountability for transaction exclusion.

Forward Inclusion Lists (FIL) [27] (Neuder et al.) is a Π_1 mechanism that enforces transaction inclusion in a *forward* manner: the inclusion list ($\mathcal{T}_{\text{req}}$) is constructed by the current block assembler and published alongside block B_s, while binding obligations apply to the subsequent block assembler β_{s+1}, who must include all listed transactions in block B_{s+1}. Since FIL does not constrain the construction of $\mathcal{T}_{\text{req}}$, the assembler of B_s retains full freedom over its contents, and omissions are not verifiable. Tsao et al. [35] introduce a first step towards accountability by adding a validity check that ensures listed transactions are includable in the next block, but the construction of obligation $\mathcal{T}_{\text{req}}$ remains unconstrained. Since the block assembler receives no reward tied to any specific transaction in $\mathcal{T}_{\text{req}}$, bribing the assembler to omit a transaction from the list incurs only a negligible additional cost. The dominant censorship cost is that of excluding transaction τ from both block B_s and B_{s+1} (see Lemma 4). A tighter characterization of per-slot censorship costs under FIL is provided in Appendix F.1. FairFIL strengthens FIL's design by introducing accountability through a protocol-defined and verifiable construction of $\mathcal{T}_{\text{req}}$, while leveraging the block reward as an intrinsic enforcement mechanism.

Fork-Choice Enforced Inclusion Lists (FOCIL) [34] (Thiery et al.), scheduled for deployment on Ethereum in 2026 [13], distributes the construction of $\mathcal{T}_{\text{req}}$ across a committee of $\kappa = 16$ validators per slot. Each committee member independently selects transactions from its local mempool view and broadcasts a partial inclusion list during slot s. The block assembler β_{s+1} is required to include all transactions appearing in any partial list, subject to block validity and capacity constraints. In contrast to FIL, transactions propagated within slot s (in particular up to three seconds before the start of slot s+1) are subject to an inclusion obligation for block assembler β_{s+1}, which classifies FOCIL as a Π_0 mechanism. FOCIL relies on a *one-of-κ-honest* assumption: a single honest committee member suffices to enforce inclusion of a transaction τ in $\mathcal{T}_{\text{req}}$, implying that an adversary must compromise all κ members to censor τ. However, as in FIL, committee members receive no reward for specific transactions they include; any positive bribe is therefore sufficient to induce omission from an individual member's partial list. Consequently, censorship resistance scales with committee size but remains vulnerable in a purely rational setting. We show in Appendix F.2 that the resulting per-slot bribery cost across the entire committee is below 2 € for a median-tip transaction in late 2025 under current Ethereum conditions (see Section 5.4).

Multiple Concurrent Proposers (MCP) [19] (Garimidi et al.) departs from the inclusion-list paradigm of FIL and FOCIL by introducing κ concurrent block assemblers (we assume $\kappa = 16$), each constructing an independent partial block. These κ partial blocks are concatenated into a single block (Π_0), forming an obligation $\mathcal{T}_{\text{req}}$, without the possibility for a single party to include additional transactions afterwards. As in FOCIL, a transaction is included in $\mathcal{T}_{\text{req}}$ if at least one proposer includes it in its partial block, again relying on a one-of-κ-honest assumption. In contrast to FIL and FOCIL, proposers receive direct rewards for transactions included in their partial blocks that are ultimately incorporated into the aggregated block. Each proposer retains at most the per-transaction tip $f^{\tau}_{\text{tip, tx}}$ for transactions in its own partial block, not considering MEV extraction. Consequently, omitting a transaction τ induces an opportunity cost equal to the forgone tip. As a result, the required per-member bribe increases by approximately the transaction tip relative to FOCIL, which is around 0.01 € at the median-tip transaction in late 2025, and is therefore only marginally higher than in FOCIL.

Auction-Based Inclusion Lists (AUCIL) [36] (Wadhwa et al.) is a Π_0 mechanism that combines committee-based construction with an explicit auction for aggregating partial inclusion lists. In the first phase, a committee of size $\kappa = 32$ constructs partial input lists. Each committee member is assigned a subset of transactions via a shared allocation mechanism, such that honest inclusion of the assigned transactions forms a correlated equilibrium. While inclusion of the assigned transactions is a rational strategy given transaction tips, compliance is not verified. In the second phase, each committee member aggregates the received partial lists and submits a bid proportional to the number of included lists, inducing competition over the final $\mathcal{T}_{\text{req}}$. The block proposer is required to select the aggregate with the highest bid and include the corresponding transaction set in its block. This design incentivizes inclusion of committee contributions, since omitting lists reduces bid strength and thus expected reward. Nevertheless, AUCIL does not provide enforceable accountability for transaction exclusion. Appendix F.4 shows that even under this mechanism, which we consider the most effective among mechanisms based solely on transaction fees, a transaction can be censored at an expected cost of approximately 3.30 € per slot under current Ethereum conditions.

Comparison. All four mechanisms share the same structural limitation: the absence of accountability in transaction selection keeps the per-slot cost of censorship low, with costs primarily determined by transaction tips and committee size. We summarize censorship costs over one block, two blocks, and ten blocks (two minutes) for each mechanism in Table 2. By enabling verifiability of transaction censorship and attribution of censorship to a specific party, misbehavior can be penalized, for instance through block invalidation and the associated loss of revenue. With FairFIL, we present a mechanism for detecting transaction censorship, attributing it to a single party, and enabling penalization. This approach leads, as indicated in Table 2, to significantly higher costs for sustained censorship in practice.

Mechanism		Committee	Censorship Costs (cumulated) 1 slot	2 slots	10 slots	Proof
Baseline		—	≈ 0.10 €	≈ 0.20 €	≈ 1 €	App. C
Prior Work		(constructing)				
FIL [27]		1	≈ 0.11 €	≈ 0.22 €	≈ 1.10 €	App. F.1
FOCIL [34]		16	≤ 1.54 €	≤ 3.08 €	≤ 15.40 €	App. F.2
MCP [19]		16	≤ 1.60 €	≤ 3.20 €	≤ 16 €	App. F.3
AUCIL* [36]		32	≤ 3.30 €	≤ 6.60 €	≤ 33 €	App. F.4
FairFIL		(verifying)				
Attack	Suppression	≈ 30 000	≈ 0.10 €	≈ 50.34 €	≈ 402 €	Sec. 5.3
Attack	**Invalidation****	≈ 30 000	≈ 0.10 €	≈ 28.30 €	≈ 141 €	Sec. 5.3

Table 2 Costs of censoring a median-tip transaction under the mechanisms proposed and surveyed in this paper. All values are based on empirical measurements from Sep. to Dec. 2025 (Section 5.4). Suppression and invalidation are the two ways in which censorship in FairFIL remains feasible at the stated cost; we describe both in detail in Section 5.3.
* For AUCIL, Wadhwa et al. [36] provide a tighter bound; in practice, bribes may be cheaper.
** Invalidating a block can be significantly more expensive due to MEV extraction.

4 Fair Forward Inclusion Lists

We present FairFIL, an accountable Π_1 censorship resistance mechanism. Block assembler β_s retains full authority over the construction of block B_s and MEV extraction, but must publish every excluded transaction in a list FairFIL$_s$ alongside the block. A committee of validators

verifies that FairFIL$_s$ is complete with respect to their mempool view; if a majority finds a discrepancy, block B_s is rejected and the assembler forfeits the full block reward $\mathcal{R}(B_s)$. The subsequent assembler β_{s+1} must include every transaction listed in FairFIL$_s$ in block B_{s+1}; any omission forfeits block reward $\mathcal{R}(B_{s+1})$. We first describe how β_s constructs FairFIL$_s$, then how validators verify its correctness and enforcement. Informally, FairFIL$_s$ is correct if it contains exactly those transactions censored from B_s that remain includable in B_{s+1}. We formalize and prove these requirements via four accountability properties in Section 5.3.

4.1 FairFIL Construction

Algorithm 1 details how block assembler β_s produces block B_s and FairFIL$_s$. At a high level, β_s performs two tasks: fulfilling the inclusion obligation from FairFIL$_{s-1}$, and publishing a complete disclosure of all transactions excluded from block B_s.

Enforcing FairFIL (lines 1–3). FairFIL$_{s-1}$ is a hard constraint on block B_s: every listed transaction must appear in B_s. Block assembler β_s integrates the sequence into the block-construction strategy (line 1); lines 2–3 verify that this constraint is satisfied. Any violation causes B_s to be rejected (see Section 4.2), forfeiting the full block reward $\mathcal{R}(B_s)$.

Constructing FairFIL (lines 4–24). The central challenge is to construct an inclusion list FairFIL$_s$ that is complete (covering every censored transaction), executable (each listed transaction must remain valid against the post-state of block B_s), and verifiable (validators must be able to independently reconstruct the same transaction set to check for undisclosed exclusions). Such an inclusion list is constructed in four steps. First, block assembler β_s considers all mempool transactions seen before the start of slot s (line 4).[2] Second, these transactions are sorted according to Fair Ordering (line 5), using the per-gas tip as primary key and the transaction hash as tie-breaker. Third, non-censored transactions – those already contained in B_s as well as those not includable in B_s under Fair Ordering – are discarded (lines 9–17). Fourth, to ensure that the resulting list remains executable for the subsequent block assembler, the censored transactions are executed on the post-B_s state (lines 18–24). Each transaction is applied in order, with the state updated after each successful execution. Since block assembler β_s assembles block B_s, the post-B_s state is available at construction time (line 1). Any transaction whose execution would violate protocol validity is discarded.

4.2 FairFIL Verification

We use the validator set V_s of slot s as the committee responsible for verifying FairFIL$_s$. Algorithm 2 refines the verification procedure performed by a validator $v' \in V_s$. A validator casts a negative vote for block B_s if any of the following checks fails. Block acceptance requires a majority of validator votes.

Check 1: Enforcement of FairFIL$_{s-1}$ (lines 1–3). Every transaction from FairFIL$_{s-1}$ must appear in block B_s. Since the committee of slot $s-1$ verified includability (Check 2), any missing transaction is unambiguous evidence of non-compliance by block assembler β_s.

[2] Without a cutoff, β_s would need to emulate every transaction in the mempool, including those persisting for months (e.g., spam or long-pending underpriced transactions). The specified 603 s window – fifty slots (600 s) plus a 3 s propagation margin (Section 5.2) – limits this to a manageable recent window.

Algorithm 1 Construction of block B_s and FairFIL$_s$ by block assembler β_s.

Input: Previous block B_{s-1}, previous FairFIL$_{s-1}$, local mempool $\text{Pool}_s^{\beta_s}$
Output: Block B_s, FairFIL$_s$
/ Block construction with FairFIL$_{s-1}$ forced in (lines 2–3). */*
/ **Note**: Assembler β_s is **not** required to follow the ordering of FairFIL$_{s-1}$. */*
1 $B_s \leftarrow$ *assembleBlock*(*optional:* $\text{Pool}_s^{\beta_s} \cup \text{XOF}$, *mandatory:* FairFIL$_{s-1}$)
2 **foreach** $\tau \in$ *FairFIL*$_{s-1}$ **do**
3 *assert*($\tau \in B_s$)
/ Construct a transaction set according to Fair Ordering. */*
4 FairOrdering $\leftarrow \{\tau \in \text{Pool}_s^{\beta_s} \mid \text{first\_seen}(\tau) \geq \text{SLOT\_START} - 603\,\text{s}\}$
5 FairOrdering $\leftarrow$ sort(FairOrdering, *prim:* $f^{\tau}_{\text{tip, gas}}$, *sec: tx hash*, *order: desc.*)
6 Censored $\leftarrow [\,]$
7 *state* $\leftarrow$ state(B_{s-1} || FairFIL$_{s-1}$) *// appending FairFIL$_{s-1}$ on state of B_{s-1}.*
8 *GasUsed* $\leftarrow$ *gas*(FairFIL$_{s-1}$)
9 **foreach** $\tau \in$ *FairOrdering* **do**
10 **if** *GasUsed* $+ 21000 > g_{\max}(B_s)$ **then**
11 **break**
 // Minimum gas consumption is 21000 gas [38]; no further transaction fits.
12 *state*$'$ $\leftarrow$ execute(τ on *state*)
13 **if** *state*$'$ *is valid* **then**
14 *state* $\leftarrow$ *state*$'$
15 *GasUsed* $\leftarrow$ *GasUsed* $+$ gas(τ)
16 **if** $\tau \notin B_s$ **then**
17 append τ to Censored
/ Determine FairFIL$_s$ by verifying includability in block B_{s+1}. */*
18 FairFIL$_s$ $\leftarrow [\,]$
19 *state* $\leftarrow$ state(B_s)
20 **foreach** $\tau \in$ *Censored* **do**
21 *state*$'$ $\leftarrow$ execute(τ on *state*)
22 **if** *state*$'$ *is valid* **then**
23 *state* $\leftarrow$ *state*$'$
24 append τ to FairFIL$_s$
25 **return** B_s, FairFIL$_s$

Check 2: Includability of FairFIL$_s$ (lines 4–8). FairFIL$_s$ must comply with Fair Ordering: transactions are sorted in descending per-gas tip order (hash as tie-breaker), and executing them sequentially on the state induced by block B_s must preserve protocol validity of block B_{s+1}. The validator emulates this execution and rejects B_s if the resulting state is invalid. A fixed ordering is essential: otherwise, β_s could reorder preceding transactions to manipulate the state such that a censored target appears non-includable – a claim validators could only refute by enumerating all permutations.

Check 3: Bounding unknown transactions (lines 9–17). FairFIL$_s$ may contain transactions unknown to a validator, either due to mempool inconsistencies or because the block assembler β_s deliberately injects non-propagated (XOF) transactions. We show in Section 5.2

Algorithm 2 Verification of FairFIL$_s$ by validator $v' \in V_s$.

Input: B_{s-1}, B_s, FairFIL$_{s-1}$, FairFIL$_s$, local mempool $\text{Pool}_s^{v'}$
Output: $\text{Vote}_{v'} \in \{\text{FairFIL}_s \text{ valid}, B_s \text{ invalid}\}$

/ Check 1: Verifying Inclusion of all previous-FairFIL transactions in B_s */*
foreach $\tau \in \mathit{FairFIL}_{s-1}$ **do**
  if $\tau \notin B_s$ **then**
    return $\mathit{Vote}_{v'} : B_s$ *invalid*

/ Check 2: Verifying self-consistency of FairFIL$_s$ */*
if $\mathit{FairFIL}_s$ *is not sorted correctly* **then**
  return $\mathit{Vote}_{v'} : B_s$ *invalid*
$\mathit{state}' \leftarrow \text{execute}(\text{FairFIL}_s \text{ on state}(B_s))$
if state' *is not valid* **then**
  return $\mathit{Vote}_{v'} : B_s$ *invalid*

/ Check 3: Bounding unknown transactions; Constructing extended mempool */*
UnknownCount $\leftarrow 0$; UnknownGas $\leftarrow 0$
$\text{ExtendedPool}_s^{v'} \leftarrow \text{Pool}_s^{v'}$
foreach $\tau \in \mathit{FairFIL}_s$ **do**
  if $\tau \notin \text{Pool}_s^{v'}$ **then**
    UnknownCount += 1;
    UnknownGas += gas(τ);
    $\text{ExtendedPool}_s^{v'} \leftarrow \text{ExtendedPool}_s^{v'} \cup \{\tau\}$
if UnknownCount $>$ threshold($\text{Pool}_s^{v'}$).count ***or***
 UnknownGas $>$ threshold($\text{Pool}_s^{v'}$).gas **then**
  return $\mathit{Vote}_{v'} : B_s$ *invalid*

/ Check 4: Check FairFIL$_s$ correctness by constructing a censorship-free block */*
$\text{FairBlock}_s \leftarrow []$
$\text{LocalFairOrdering} \leftarrow \{\tau \in \text{ExtendedPool}_s^{v'} \mid \tau \in \text{FairFIL}_s \vee 3\,\text{s} \leq \text{SLOT\_START} - \text{first\_seen}(\tau) \leq 600\,\text{s}\}$
$\text{LocalFairOrdering} \leftarrow \text{sort}(\text{LocalFairOrdering},\ \textit{prim: } f^{\tau}_{\text{tip, gas}},\ \textit{sec: tx hash},\ \textit{order: desc.})$
$\mathit{state} \leftarrow \text{state}(B_{s-1} \,||\, \text{FairFIL}_{s-1})$ *// appending FairFIL$_{s-1}$ on state of B_{s-1}.*
$\mathit{GasUsed} \leftarrow \mathit{gas}(\text{FairFIL}_{s-1})$
foreach $\tau \in \mathit{LocalFairOrdering}$ **do**
  if $\mathit{GasUsed} + 21000 > g_{\max}(B_s)$ **then**
    break
  $\mathit{state}' \leftarrow \text{execute}(\tau \text{ on } \mathit{state})$
  if state' *is valid* **then**
    $\mathit{state} \leftarrow \mathit{state}'$
    $\mathit{GasUsed} \leftarrow \mathit{GasUsed} + \text{gas}(\tau)$
    append τ to FairBlock_s
foreach $\tau \in \mathit{FairBlock}_s$ **do**
  if $\tau \notin \mathit{FairFIL}_s \wedge \tau \notin B_s$ **then**
    return $\mathit{Vote}_{v'} : B_s$ *invalid*
return $\mathit{Vote}_{v'} : \mathit{FairFIL}_s$ *valid*

that minor mempool inconsistencies can occur, particularly at home-staking validators; highly optimized builders, operating multiple nodes, are expected to maintain a complete mempool view. Injected XOF poses two problems. First, any XOF transaction in FairFIL$_s$ must be included in block B_{s+1}, allowing block assembler β_s to extract MEV across two slots and depriving β_{s+1} of both block space and the opportunity to extract this MEV itself – a dynamic β_s can exploit strategically. Second, a central objective of accountability is that actions within FairFIL remain verifiable for users; injected XOF undermines this, as such transactions appear in FairFIL$_s$ without prior public visibility and can displace legitimate mempool transactions, as demonstrated in Table 1. Since validators cannot distinguish injected XOF from mempool inconsistencies, we define a private, per-validator threshold on the count and cumulative gas of unknown transactions in FairFIL$_s$; exceeding the threshold causes B_s to be rejected. We specify the threshold in Section 5.2 and argue that the size required to absorb mempool inconsistencies is too small for β_s to consume substantial block space or extract meaningful MEV. Unknown transactions within the threshold bounds are added to an extended mempool to preserve dependency chains between known and unknown transactions (line 15).

Check 4: Accountability (lines 18–33). To verify that β_s accounted for every exclusion, the validator applies Fair Ordering to the extended mempool and constructs a local 'censorship-free' reference block FairBlock$_s$. Every transaction in FairBlock$_s$ must appear either in block B_s (included) or in FairFIL$_s$ (censored by β_s). Any transaction absent from both reveals an undisclosed exclusion and causes the validator to vote against B_s.

5 Analysis of FairFIL

In this section, we first demonstrate deployability by evaluating FairFIL on Ethereum mainnet (Section 5.1) and validate the key assumption of consistent mempools empirically (Section 5.2). Then, we prove that non-compliance with FairFIL causes block invalidation, thereby establishing $(\mathcal{M}, 1)$-censorship resistance, where $\mathcal{M}$ is approximately equal to the block reward (Section 5.3). Section 5.4 quantifies censorship costs across all mechanisms.

5.1 Empirical Evaluation

We implement FairFIL and emulate its operation on 214 600 Ethereum mainnet blocks from September 2025, measuring the sizes of the resulting per-slot inclusion lists and the number of censored transactions forced into the subsequent block. To identify censored transactions, we reconstruct a censorship-free reference block for each slot and compare it to the actual Ethereum mainnet block B_s, then construct FairFIL$_s$ according to Algorithm 1.

Method. We use transactions from Flashbots' Mempool Dumpster [16], a public archive of publicly propagated transactions collected by multiple geographically distributed nodes. For each block B_s, we extract every transaction first observed between 3 s and 600 s before the start of slot s and sequence them according to Fair Ordering. We execute this sequence using a modified version [4] of Foundry's Anvil [17], adapted to replay a specified transaction sequence against a given historical state. We skip transactions that are not validly includable, as they are not censored in block B_s. The historical state corresponds to the post-state of block B_{s-1} and was obtained from a self-operated Ethereum execution-layer node [28].

Results. While most blocks are censorship-free, we observe an average of 8.3 censored transactions per block across all observed blocks, amounting to 1.78 million transactions censored by at least one slot during September 2025. Of the 214 600 blocks examined, 81 411 blocks (38 %) yielded a non-empty FairFIL$_s$; averaged over all blocks, a FairFIL$_s$ contains 7.3 transactions. Figure 1 shows the sizes of FairFIL$_s$, grouped by block assembler. Each non-empty FairFIL$_s$ consumed a median of 63 209 gas in the subsequent block (95th percentile (P95): 6 057 234 gas), corresponding to 0.1 % of the block's gas budget (P95: 13.3 %). Inclusion of the transactions in FairFIL$_s$ succeeded in every case, as we formally prove in Lemma 8. We validated robustness by repeating the analysis with the mempool of our own Ethereum node and obtained nearly identical results. Of all transactions censored in slot s, 87.4 % were still includable in block B_{s+1} and enforced by FairFIL$_s$; the remaining 12.6 % were no longer includable. For 49.1 % of these non-includable transactions, the provided base fee fell below block B_{s+1}'s base fee. Since base fees rise by at most 12.5 % between two consecutive blocks [8], this non-includability can be prevented by accounting for this worst-case increase upfront in the user's base-fee bid. A further 50.6 % had been replaced by the user, and the remaining 0.3 % were prevented by other user-initiated state changes. Constructing the FairFIL$_s$ takes at most 0.388 s per slot in our unoptimized, single-threaded implementation on an AMD EPYC 4564P. This is well below the 4 s attestation deadline, indicating that construction cost is not a limiting factor for the timeliness requirements imposed by Ethereum's current slot anatomy.

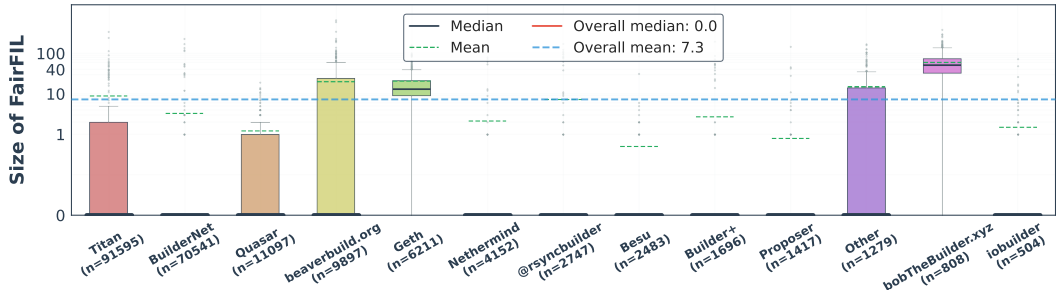

Figure 1 Size of FairFIL (in number of transactions) per block between blocks 23 264 600 and 23 479 200 (September 2025), grouped by block assembler. Note the logarithmic y-axis and the number n of blocks proposed by each assembler during the observation period. "Proposer" reflects block assemblers that disclose no identifying information. Block assemblers with fewer than 500 proposed blocks during the observation period are aggregated into "Other".

5.2 Mempool Consistency

Both the operation and the analysis of FairFIL's censorship-resistance properties assume that all participants share a sufficiently consistent mempool view. We now revisit this assumption using the September 2025 mempool dataset of the Decentralized Systems Lab at Yale University [11], a data source distinct from the one used in Section 5.1 to guard against source-specific artifacts. In particular, we need to provide evidence that block assemblers can observe every transaction the majority of validators deems relevant under Fair Ordering, and that this majority observes those transactions within the required time window. We show that both hold sufficiently well for FairFIL: block assemblers can be expected to observe all transactions propagated through Ethereum's P2P network in a timely manner, and in every block, a vast majority of validators would have voted correctly on FairFIL compliance.

Measurement setup. The dataset provided in [11] contains mempool observations from three nodes geographically distributed across the United States and Germany. We combine this dataset with mempool data from our own Ethereum node, located in Karlsruhe (Germany). Each node logs the first-observation timestamp of a transaction received through Ethereum's P2P layer. For every transaction contained in the censorship-free blocks (cf. Section 5.1), we take the earliest first-observation timestamp across all four nodes as the transaction's propagation origin and then measure how many of the remaining nodes observed the transaction within the subsequent 3 s.

In this analysis, we do not consider transactions that were not includable in the next possible block, since spam transactions (e.g., severely underpriced ones) would otherwise distort the measurement. The window of three seconds is motivated by prior measurements showing that block assemblers typically begin block construction about 3 s before slot start. Overall, the measurement covers $n = 19\,609\,591$ transactions from September 2025.

Results. In 96.12 % of all blocks, all four monitors observed every transaction that should appear in a censorship-free block. Moreover, for every block, at least two of the four nodes observed all such transactions. When deviations occurred, they were typically minimal: for each individual node, about 2 % of blocks were missing at most a single required transaction. For the remaining 1.88 % of blocks, closer inspection suggests that the deviations are primarily caused by temporary disruptions at individual nodes, since missed transactions typically occur in bursts across multiple consecutive blocks. Propagation latency across the four nodes, defined as the time between the first observation at any node and the first observation at the last receiving node, was below 0.468 s for 95 % of transactions. Figure 3 in Appendix B shows the resulting per-node consistency distribution. Two quantitative observations follow.

▶ **Observation 6** (Single-node coverage). *A single reachable node observes at least 90 % of the public transactions that Fair Ordering requires for inclusion in block B_s, within 3 s of first observation by any of the monitoring nodes.*

▶ **Observation 7** (Multi-node coverage). *A block assembler that merges the mempool views of three independent nodes observes every public transaction that Fair Ordering requires for inclusion in block B_s; using two nodes already covers at least 99.79 % of these transactions.*

Interpretation. The data suggest that this level of consistency is sufficient for FairFIL. These results do not imply that mempool consensus is trivial. Instead, they provide evidence that the transaction set relevant to FairFIL per slot is likely observable by both a well-provisioned builder and the validator majority within the required time window, without requiring agreement on temporal ordering. Under enshrined PBS [10], expected in mid-2026, proposers obtain protocol-level access to a specialized builder market in which builders operate as highly optimized infrastructure rather than as individual home servers. Under these conditions, a rational builder can be expected to operate multiple geographically distributed nodes and, by Observation 7, not to miss any transaction that has been known to a majority of nodes for at least 3 s. Regarding enforcement, the same does not hold for validators running a single node, so FairFIL includes an adaptive, validator-local XOF threshold (cf. Algorithm 2, Check 3) to mitigate occasional false votes due to incomplete mempool views. The threshold should be unknown to the block assembler and could be designed to self-adjust: rising on repeated disagreement with the majority, falling otherwise. Empirically, a majority of validators observes all FairFIL-relevant transactions in all cases (Observation 7). Our evaluation of the adaptive threshold shows that it remains equal to zero for most validators and close to zero for the remainder across all blocks.

5.3 Analyzing FairFIL's Censorship Resistance

We introduce four properties that jointly characterize an accountable mechanism and imply censorship resistance against all protocol-deviating attacks. We prove that FairFIL satisfies each property under the assumption of a bribing budget $\mathcal{M} \leq \min(\mathcal{R}(B_s), \mathcal{R}(B_{s+1}))$: any violation of an accountability property invalidates a block either at construction (rejecting block B_s) or at enforcement time (rejecting block B_{s+1}). Property 1 ensures that the obligation to include the set of required transactions $\mathcal{T}_{\text{req}}$ imposed by FairFIL can be fulfilled, so failing to include $\mathcal{T}_{\text{req}}$ is unambiguous non-compliance. Property 2 requires every censored transaction to appear in $\mathcal{T}_{\text{req}}$; Property 3 restricts $\mathcal{T}_{\text{req}}$ to publicly observed transactions. Property 4 requires that the obligated assembler has no rational incentive to deviate from the obligation. Enforcement rests on majority voting of V_s on Checks 1–4 (Algorithm 2); reversing this verdict requires bribing a majority of V_s at a cost exceeding the full block reward $\mathcal{R}(B_s)$. Throughout, we assume the majority of validators votes correctly.

▶ **Property 1** (Includability). *For each transaction $\tau \in \mathcal{T}_{\text{req}}$, the block assembler in slot $s+n$ can include τ in block B_{s+n} at the position prescribed by Π_n, while preserving block validity.*

A foundational requirement of accountability is the distinction between deliberate non-compliance and circumstances that prevent compliance. Property 1 establishes this distinction by admitting only obligations that the corresponding block assembler can fulfill. Without this property, the assembler could populate $\mathcal{T}_{\text{req}}$ with transactions that consume list capacity but cannot be included in the enforcement block, thereby preventing inclusion of the target transaction. Prior work [36] refers to this property as "unconditional".

▶ **Lemma 8.** *FairFIL satisfies Property 1: every transaction in FairFIL$_s$ is includable in block B_{s+1} at the position prescribed by FairFIL$_s$ without violating the validity of B_{s+1}. Any transaction failing this condition would cause block B_s to be rejected.*

Proof. By Check 2 of Algorithm 2, the includability of FairFIL$_s$ requires that executing the entire transaction sequence of FairFIL$_s$ on top of B_s's induced state yields a valid state. Since every operation in Ethereum is deterministic [6], all validators compute the same state and apply the same FairFIL$_s$, so the execution outcome is identical across validators. Because transactions in a sequential execution do not retroactively invalidate earlier transactions – each transaction's validity depends only on the state induced by its predecessors – every prefix of FairFIL$_s$ executes validly on top of B_s's induced state. In particular, the prefix ending in τ executes without violating validity. A majority of validators in V_s has verified this condition and accepted the block; otherwise B_s is rejected. Therefore, β_{s+1} can include this prefix at least at the beginning of B_{s+1} without violating the validity of the protocol. ◀

▶ **Property 2** (Accountable listing). *Let τ be a transaction observed by a majority of committee members by a specific point in time. Following Fair Ordering: if τ is includable in block B_{s+n}, then $\tau \in \mathcal{T}_{\text{req}}$, unless including τ would displace a higher-tipped transaction τ'.*

An enforceable obligation does not establish accountability if the content of the obligation cannot be verified. Property 2 extends accountability to the construction of $\mathcal{T}_{\text{req}}$: from a consistent mempool view, any party can determine which transactions Fair Ordering mandates, thereby making any omission attributable to the corresponding assembler.

▶ **Lemma 9.** *FairFIL satisfies Property 2: every transaction τ that (i) is observed at least 3 s before the start of slot s by a majority of V_s, (ii) is includable in both blocks B_s and B_{s+1},*

and (iii) would not displace any transaction τ' with $f^{\tau'}_{\text{tip, gas}} > f^{\tau}_{\text{tip, gas}}$, is included in either block B_s or FairFIL$_s$. Any omission causes block B_s to be rejected.

Proof. Assume, for contradiction, that τ satisfies conditions (i)–(iii) but $\tau \notin B_s \cup \text{FairFIL}_s$. Let $W \subseteq V_s$ denote the set of validators that observe τ at least 3 s before the start of slot s; by (i), $|W| > \frac{|V_s|}{2}$. Consider any validator $v \in W$ executing Check 4 of Algorithm 2. Since $\tau \in \text{Pool}^v_s \subseteq \text{ExtendedPool}^v_s$ (Check 3), τ is contained in the local Fair Ordering reconstructed by v. By (iii), no higher-tipped transaction displaces τ, so the iteration over LocalFairOrdering reaches τ; by (ii), executing τ yields a valid state, so τ is admitted into FairBlock$_s$. Since $\tau \in \text{FairBlock}_s$ but $\tau \notin B_s \cup \text{FairFIL}_s$, validator v detects undisclosed transaction censorship and votes against B_s. This argument applies to every validator in W, so a majority of V_s, since $|W| > \frac{|V_s|}{2}$, rejects B_s, and block assembler β_s forfeits the entire block reward $\mathcal{R}(B_s)$. Compensating this loss would require a bribe exceeding $\mathcal{R}(B_s) \geq \mathcal{M}$, contradicting the budget restriction and thus the assumption $\tau \notin B_s \cup \text{FairFIL}_s$. ◀

▶ **Property 3** (Majority observability). *The obligation T_{req} must contain only transactions observed by a majority of committee members.*

While Property 2 prevents omission of eligible transactions, a rational assembler may also exploit the obligation by injecting own transactions that were not publicly broadcast beforehand. This undermines accountability, since the construction of T_{req} is no longer publicly verifiable, and may displace censored transactions. Table 1 illustrates the consequence: placing τ_{private} in T_{req} leaves insufficient capacity for the censored τ_4. Property 3 rules out such injections by restricting T_{req} to transactions observed by a majority of V_s.

▶ **Lemma 10.** *Under Ethereum's empirical mempool consistency (Section 5.2), FairFIL satisfies Property 3: every transaction τ contained in FairFIL$_s$ is observed by a majority of V_s; otherwise, a majority of V_s rejects block B_s.*

Proof. By Section 5.2, a majority of validators set the XOF threshold in Check 3 of Algorithm 2 to zero; we assume this throughout. Suppose, for contradiction, that some transaction $\tau \in \text{FairFIL}_s$ is not observed by a majority of V_s. If all validators share the same mempool view, then $\tau \notin \text{Pool}^v_s$ for every validator $v \in V_s$. Check 3 therefore marks τ as unknown at every validator, and with threshold zero, every $v \in V_s$ rejects block B_s. If mempools are heterogeneous, then there exists a majority $W \subseteq V_s$ with $|W| > |V_s|/2$ such that $\tau \notin \text{Pool}^v_s$ for every $v \in W$. For each $v \in W$, Check 3 marks τ as unknown, and with threshold zero, v rejects B_s. Hence a majority of V_s rejects B_s. In either case, block assembler β_s forfeits the full block reward $\mathcal{R}(B_s) \geq \mathcal{M}$. Compensating this loss would require a bribe exceeding $\mathcal{R}(B_s)$, contradicting the budget bound. ◀

▶ **Property 4** (Rational builder compliance). *The block assembler required to include transaction set $\mathcal{T}_{\text{req}}$ in block B_s does so if the assembler behaves rationally.*

Properties 1–3 make the correctness of transaction set $\mathcal{T}_{\text{req}}$ publicly verifiable; accountability, however, requires that this obligation is also enforced. If non-compliance were the rational choice for the obligated block assembler, the assembler would deviate under the rationality model, and the censorship resistance mechanism would fail at enforcement. Compliance with $\mathcal{T}_{\text{req}}$ must therefore constitute the rational choice for the obligated assembler.

▶ **Lemma 11.** *FairFIL satisfies Property 4: a rational (or altruistic) block assembler β_{s+1} includes every transaction $\tau \in \text{FairFIL}_s$ in block B_{s+1}. Any omission of a listed transaction from B_{s+1} causes block B_{s+1} to be rejected.*

Proof. Assume, for contradiction, that a rational block assembler β_{s+1} does not include some transaction $\tau \in \text{FairFIL}_s$ in block B_{s+1}. By Lemma 8, the omission of τ is a deliberate choice by β_{s+1}, not a consequence of infeasibility. At slot $s{+}1$, every validator $v \in V_{s+1}$ runs Check 1 of Algorithm 2, which verifies that every transaction in FairFIL_s appears in block B_{s+1}. Since, by assumption, $\tau \notin B_{s+1}$, Check 1 fails for every $v \in V_{s+1}$; the block is rejected and β_{s+1} forfeits the entire block reward $\mathcal{R}(B_{s+1})$. Compensating this loss would require a bribe exceeding $\mathcal{R}(B_{s+1}) \geq \mathcal{M}$, contradicting the budget restriction. Any lower bribe would contradict the rationality assumption of β_{s+1}. ◀

The preceding lemmas establish that any protocol-deviating censorship attempt – whether by omitting a required transaction from FairFIL_s, injecting unauthorized transactions into FairFIL_s, or failing to enforce the resulting inclusion obligation – causes block rejection at the cost of $\mathcal{R}(B_s)$ (construction failure) or $\mathcal{R}(B_{s+1})$ (enforcement failure).

▶ **Lemma 12.** *If censorship of transaction τ for more than one slot requires a deviation from FairFIL, then FairFIL is $\big(\min(\mathcal{R}(B_s), \mathcal{R}(B_{s+1})) + c_{\text{bribe}},\ 1\big)$-censorship-resistant. Equivalently, by deviating from FairFIL, two-slot censorship of transaction τ requires invalidating at least one of the blocks B_s or B_{s+1}.*

The proof is given in Appendix D. Let us now focus on protocol-compliant censoring, i.e., attacks that comply with FairFIL but try to suppress a transaction by other means. In a *transaction suppression attack*, adversary $\mathcal{A}$ acts as an ordinary user and propagates high-tip filler transactions through the network to displace τ below the gas-limit cutoff of the Fair Ordering. We show that, for nearly all transactions, this attack is more expensive than block invalidation.

Intuition. Acting as an ordinary user, adversary $\mathcal{A}$ propagates transactions through the network such that they appear in regular mempools, where FairFIL treats them equivalently to any other transaction. By propagating a filler transaction τ' ranked strictly above τ under Fair Ordering – for simplicity, we assume a single τ' suffices and disregard the per-transaction gas cap [3] – $\mathcal{A}$ displaces transaction τ below the block's gas limit $g_{\max}(B_s)$. Transaction τ is consequently not considered censored and therefore not included in FairFIL_s; block assembler β_{s+1} is thus not obligated to include τ in block B_{s+1}, allowing τ to be censored at low cost in B_{s+1}. The attack is rule-conforming: FairFIL is verified and enforced as specified. In the following, we derive the $(\mathcal{M}, \sigma)$-censorship resistance that FairFIL achieves against this attack.

Formalization. We write $\tau' \succ \tau$ if transaction τ' ranks strictly above transaction τ under Fair Ordering, that is, if $f^{\tau'}_{\text{tip, gas}} > f^{\tau}_{\text{tip, gas}}$, or if $f^{\tau'}_{\text{tip, gas}} = f^{\tau}_{\text{tip, gas}}$ and $\text{hash}(\tau') < \text{hash}(\tau)$. Let $G_{\text{above}}(\tau, s)$ be the cumulative gas of all mempool transactions ranked above transaction τ at slot s, and define the residual gas capacity $\Delta(\tau, s)$ above τ as the largest amount of additional gas that can be inserted above τ in Fair Ordering while τ still fits within block B_s:

$$\Delta(\tau, s) \;=\; \max\big(0,\ g_{\max}(B_s) - G_{\text{above}}(\tau, s) - \text{gas}(\tau)\big).$$

To suppress τ, adversary $\mathcal{A}$ must inject filler gas strictly exceeding $\Delta(\tau, s)$ in transactions satisfying $\tau' \succ \tau$, at a per-gas cost of at least $f^*(\tau) := f^{\tau}_{\text{tip, gas}} + f^{\tau'}_{\text{base, gas}}$. Matching $f^{\tau}_{\text{tip, gas}}$ rather than strictly exceeding it suffices, since $\mathcal{A}$ can outrank τ via the hash tie-breaker.

▶ **Lemma 13** (Suppression cost). *FairFIL is at most $\big(f^*(\tau) \cdot \Delta(\tau, s),\ 1\big)$-censorship-resistant for any transaction τ, propagated before slot s.*

Proof Sketch. An adversary who wants to exclude transaction τ for more than a single slot must first prevent τ from qualifying as censored in slot s – otherwise, β_{s+1} would be obligated to include τ via FairFIL$_s$. This requires injecting a publicly broadcast filler transaction τ' whose gas usage exceeds the residual gas capacity of block B_s, at a per-gas cost of at least the total per-gas transaction fee of τ, so that τ no longer fits under Fair Ordering and is not classified as censored. With no inclusion obligation in B_{s+1}, the adversary can then exclude τ from B_{s+1} at the Ethereum baseline cost $\mathcal{C}_\tau^{\text{Eth}}$. The full proof is given in Appendix E. ◀

Unlike prior mechanisms, the per-slot cost of censorship under FairFIL is not uniform, but can vary across slots. We derive this fact as follows. At slot s, no inclusion obligation applies to transaction τ, so censoring τ from block B_s costs only the Ethereum baseline $\mathcal{C}_\tau^{\text{Eth}}$. To extend censorship beyond a single slot, $\mathcal{A}$ must either prevent τ from entering FairFIL$_s$ via the suppression attack or invalidate one of the affected blocks. For the transaction suppression attack, $\mathcal{A}$ propagates a filler transaction τ' in the same slot as τ, w.l.o.g. slot s, such that τ' is included by block assembler β_s in block B_s. The block B_s is thereby fully utilized, leaving τ neither included nor classified as censored, and thus excluded from FairFIL$_s$. Block assembler β_{s+1} is therefore not obligated to include τ in block B_{s+1}, allowing adversary $\mathcal{A}$ to achieve censorship of B_{s+1} by paying $\mathcal{C}_\tau^{\text{Eth}}$. Under this payment, β_{s+1} is then required to include τ in FairFIL$_{s+1}$, so $\mathcal{A}$ must again prevent inclusion of τ in FairFIL$_{s+1}$. This coincides with the displacement required for FairFIL$_s$ and is necessary in every slot in which τ is to be censored for at least two blocks. Thus, the first and last slot in which τ is censored are comparatively cheap, while all intermediate slots require repeated displacement from the preceding FairFIL. The per-slot cost is therefore

$$\mathcal{C}_\tau \;=\; \big[\mathcal{C}_\tau^{\text{Eth}},\; f^*(\tau)\cdot\Delta(\tau,s),\; f^*(\tau)\cdot\Delta(\tau,s+1),\ldots,\mathcal{C}_\tau^{\text{Eth}}\big].$$

We show in Section 5.4 that, in late 2025, most Ethereum transactions already pay tips sufficient for suppression to cost nearly twice as much as block invalidation; even the mandatory base fee alone exceeds the block invalidation cost.

For block invalidation, once a block B_s is invalidated, the subsequent block assembler β_{s+1} carries no inclusion obligation, so τ can be censored from block B_{s+1} at Ethereum baseline cost $\mathcal{C}_\tau^{\text{Eth}}$. However, the censored transaction must then appear in FairFIL$_{s+1}$, requiring another block invalidation to sustain censorship in block B_{s+2}. This yields an alternating cost pattern between baseline censorship costs and block invalidation. The resulting per-slot cost pattern therefore takes one of two forms, or any combination of the underlined parts,

$$\mathcal{C}_\tau \;=\; \big[\underline{\mathcal{C}_\tau^{\text{Eth}},\; \mathcal{R}(B_{s+1}) + \epsilon + c_{\text{bribe}}},\ldots\big] \quad\text{or}\quad \mathcal{C}_\tau \;=\; \big[\underline{\mathcal{R}(B_s) + \epsilon + c_{\text{bribe}}\,,\; \mathcal{C}_\tau^{\text{Eth}}},\ldots\big],$$

depending on whether adversary $\mathcal{A}$ invalidates B_{s+1} (enforcement failure) or B_s (construction failure). Together, these results determine the $(\mathcal{M},1)$-censorship resistance of FairFIL.

5.4 Cost of Censorship

We now instantiate the bounds of Lemmas 12 and 13 with empirical Ethereum data to estimate the actual costs of censorship under FairFIL. Table 2 in Section 3 summarizes all results, including surveyed prior mechanisms with the instantiations described below.

Empirical Baseline Parameters. To determine the median transaction parameters, we use data from 194 million transactions in blocks 23 264 566–24 136 052 (Sep.–Dec. 2025). Figure 2 shows costs for transaction tips, transaction fees, and block tips at a reference price of 2500 € per Ether, separated into transactions propagated publicly and all transactions

(a) Transaction tip **(b)** Transaction fee **(c)** Block tip **(d)** % of XOF per block

Figure 2 Historical transaction and block data ($n = 194\,051\,211$ transactions in blocks 23 264 566–24 136 052, Sep.–Dec. 2025) from the Ethereum mainnet. (a) the transaction tip ($f^{\tau}_{\text{tip, tx}}$) paid to the assembler (in €); (b) the total transaction fee ($f^{\tau}_{\text{tip, tx}} + f^{\tau}_{\text{base, tx}}$) paid by the user (in €); (c) the block reward from tips (in €); (d) the per-block fraction of transactions that had not been observed in our mempool dataset [11]. We use a constant exchange rate of 2500 € per Ether.

within the block, as well as the per-block fraction of transactions not observed in our mempool dataset. A public transaction in this period consumes a median of 40 400 gas, pays a median per-transaction tip of $f^{\tau}_{\text{tip, tx}} \approx 0.01$ € to the block assembler, and incurs a total fee of approximately 0.05 € paid by the user. A block assembler collects a median of $\mathcal{R}_{\text{pub}}(B_s) \approx 9.09$ € in tips per block from public transactions. At the median, the included public transactions require approximately 7.9×10^6 gas, far below the block gas limit of 45×10^6 gas to 60×10^6 gas in this period (45×10^6 gas in September 2025). Private transactions (≈ 44 % of the median block's transactions) raise the assembler's revenue by a factor of 3.1, yielding a median block reward of $\mathcal{R}(B_s) \approx 28.11$ €. Based on these data and corresponding to our bribing contract implementation, we set $c_{\text{bribe}} = 0.08$ € (≈ 85 000 gas at base fees)[3] and $\epsilon = 0.01$ €. The chosen deviation incentive ϵ is small in absolute terms and, in our view, only meaningful as an incentive for actors that accumulate bribes across many slots, such as operators of many validators [14]. We nevertheless adopt this conservative value to ensure that the protocol resists any short-term rational deviations.

Quantifying Censorship Costs. To censor a transaction τ, adversary $\mathcal{A}$ must either invalidate a block (Lemma 12) or suppress the transaction (Lemma 13). Empirically, invalidating a block results in a loss of $\mathcal{R}(B_s) \approx 28.11$ €; accepting a bribe is therefore the rational decision for a payment of ≈ 28.20 €. In practice, this value is likely higher, since a block assembler typically extracts profitable MEV beyond the tips. To determine the cost of displacing τ from the residual gas capacity in Fair Ordering, we consider τ at the lowest rank under Fair Ordering and quantify the cost of suppressing such a transaction. Empirically, the adversary needs a median of ≈ 37 100 000 gas across all observed blocks to suppress τ. These are tip-less transactions; at $f^*(\tau) \approx \frac{0.038\ €}{40\,400\ \text{gas}}$, filling this capacity costs approximately 34.9 €. The same calculation for a transaction with a median tip yields 43 000 000 gas and a cost of 50.24 €. Suppression is therefore, in most cases, more expensive than block invalidation.

The corresponding costs for n-slot censorship, along with a comparison to prior proposals, are reported in Table 2. Note that whereas prior proposals prevent censorship in the earliest possible slot, censoring a transaction under FairFIL remains cheap in the first

[3] We do not implement a fully trustless bribing setup; instead, we rely on the correct operation of an oracle. The reported 85 000 gas correspond to the average per-participant cost of bribing 16 participants, with significant variation depending on the number of participants. The average cost per bribe increases for fewer than 16 participants and decreases for larger sets of participants.

slot, with substantially higher costs arising at alternating slots thereafter. For example, to censor a transaction for two minutes (ten slots), Ethereum's upcoming censorship resistance mechanism [13], FOCIL, requires ten times the per-slot bribery cost (total median: 15.40 €), whereas FairFIL requires five block invalidations interleaved with five baseline Ethereum censorship bribes (total median: 141 €).

6 Discussion

6.1 On Fair Ordering and Π_1 Mechanisms

FairFIL uses a tip-based reference ordering over publicly observed transactions rather than a receive-time-based fairness notion such as the Aequitas family [24]. Imposing such an ordering on transactions – together with preventing private communication channels – would arguably come closer to the ideal of a fair smart-contract platform. In Ethereum, however, the effects of such constraints on MEV extraction and participant behavior remain, to the best of our knowledge, insufficiently understood. This uncertainty matters because private transactions play a major role in practice. In our measurement period from September to December 2025, they consumed 2.86× as much gas as public transactions. Consistent with prior work [22, 40], this indicates extensive MEV extraction and suggests that private transactions may displace public ones when blocks are fully utilized. These observations motivate our choice of a Π_1 mechanism, although FairFIL could also be instantiated as a Π_0 mechanism with minor timing adjustments. A Π_0 mechanism would constrain the current block assembler with respect to all transactions includable in the current slot. Table 1 illustrates the tension by example: enforcing the Fair Ordering over public transactions in full would leave insufficient capacity for the privately submitted transaction τ_{private}. Relaxing to Π_1 does not fully eliminate this concern, as a large FairFIL$_s$ may still burden the next assembler. Empirically, however, this obligation is much smaller than what a Π_0 mechanism would impose on the current block. Even at the 95th percentile, the inclusion obligation induced by FairFIL$_s$ occupies only about one seventh of the subsequent block's gas capacity, whereas a Π_0 design would consume roughly half of the current block's capacity. Given the gas-consumption skew toward private transactions, an accountable Π_0 mechanism would therefore already constrain MEV extraction – an effect that warrants further study. We therefore argue that FairFIL's guarantee – either time-bounded transaction inclusion or censorship costs at the level of one invalidated block – meaningfully improves network reliability while remaining compatible with Ethereum's current incentive model and slot anatomy.

6.2 Absence of Altruism

Our behavior model assumes the absence of altruistic participants. However, prior work [32] has shown their existence and estimates their prevalence in early 2025 to be at most 1.55 % of validators. Relative to the current state of Ethereum, this implies that in roughly every 67th block, an adversarial bribe fails, even if accepting the bribe would be the rational decision for the block assembler. In particular, in an uncongested network, this yields an effective censorship delay of 13.4 minutes, which we consider insufficient for time-critical applications. For construction-based mechanisms (cf. Section 3), this further implies that in committees of size $\kappa = 128$ under a one-of-κ-honest assumption, it is likely that at least one altruistic validator is present. However, at such scales, the influence of an individual committee member becomes small, and even then inclusion of a censored transaction cannot be guaranteed. As a result, users have no guarantee that their transaction will either be included or that

exclusion requires a high-priced bribe by an adversary. Like AUCIL, FairFIL does not rely on altruistic behavior, but instead assumes predominantly rational participants while maximizing the bribe required for successful censorship.

7 Conclusion

Accountability for exclusion decisions – requiring every block assembler to publicly disclose all censored transactions in a verifiable and attributable manner – can substantially raise the cost of censoring a transaction in Ethereum. We established this claim through three contributions. First, we formalized accountability via four properties whose joint satisfaction anchors censorship costs to the full block reward, rather than to the per-transaction tip or committee size as in prior approaches. Empirically, this yields an order-of-magnitude increase relative to prior work such as FOCIL, Ethereum's upcoming censorship resistance mechanism. Second, we presented FairFIL as an accountable Π_1 mechanism and proved that, under a fully rational participant model, a deviation from FairFIL either causes block invalidation or incurs costs of comparable magnitude. Further, FairFIL requires no changes to Ethereum's existing incentive model. Third, we implemented FairFIL and empirically evaluated it on 214 600 Ethereum mainnet blocks: mempool consistency across validators appears to suffice for FairFIL operation, two-slot censorship costs rise from 3.08 € (FOCIL) to 28.30 € (FairFIL), and the median inclusion obligation imposed on the subsequent block remains small, leaving the assembler's MEV-extraction freedom largely intact.

References

1 Amitanand S. Aiyer, Lorenzo Alvisi, Allen Clement, Mike Dahlin, Jean-Philippe Martin, and Carl Porth. BAR Fault Tolerance for Cooperative Services. In *Proceedings of the 20th ACM Symposium on Operating Systems Principles (SOSP)*, SOSP '05, pages 45–58, New York, NY, USA, 2005. ACM. doi:10.1145/1095810.1095816.

2 Maryam Bahrani, Pranav Garimidi, and Tim Roughgarden. Centralization in Block-Building and Proposer-Builder Separation. In *Financial Cryptography and Data Security (FC 2024)*, pages 331–349. Springer, 2025. doi:10.1007/978-3-031-78676-1_19.

3 Tim Beiko. EIP-7825: Transaction Gas Limit Cap. Ethereum Improvement Proposal, 2024. Accessed: 2026-05-09. URL: https://eips.ethereum.org/EIPS/eip-7825.

4 Nils Henrik Beyer, Patrick Spiesberger, and Foundry. Foundry's Anvil, adapted for MCP and FairFIL. https://github.com/kit-dsn/foundry_fairfil, 2026. GitHub repository. Accessed: 2026-05-09.

5 Pedro Braga, Georgios Chionas, Piotr Krysta, Stefanos Leonardos, Georgios Piliouras, and Carmine Ventre. MEV Sharing with Dynamic Extraction Rates. In *Proceedings of the Workshop on Decentralized Finance and Security*, page 1–10, New York, NY, USA, 2024. Association for Computing Machinery. doi:10.1145/3689931.3694910.

6 Vitalik Buterin. Ethereum: A Next-Generation Smart Contract and Decentralized Application Platform. White paper. Accessed: 2026-05-09, 2014. URL: https://ethereum.org/content/whitepaper/whitepaper-pdf/Ethereum_Whitepaper_-_Buterin_2014.pdf.

7 Vitalik Buterin. The Problem of Censorship, 2015. Accessed: 2026-05-09. URL: https://blog.ethereum.org/2015/06/06/the-problem-of-censorship.

8 Vitalik Buterin, Eric Conner, Rick Dudley, Matthew Slipper, Ian Norden, and Abdelhamid Bakhta. EIP-1559: Fee Market Change for ETH 1.0 Chain. Ethereum Improvement Proposal, 2019. Accessed: 2026-05-09. URL: https://eips.ethereum.org/EIPS/eip-1559.

9 Philip Daian, Steven Goldfeder, Tyler Kell, Yunqi Li, Xueyuan Zhao, Iddo Bentov, Lorenz Breidenbach, and Ari Juels. Flash Boys 2.0: Frontrunning in Decentralized Exchanges, and Miner Extractable Value, and Consensus Instability. In *Proceedings of the 2020 IEEE Symposium on Security and Privacy (SP)*, pages 910–927. IEEE, 2020. doi:10.1109/SP40000.2020.00040.

10 Francesco D'Amato, Barnabé Monnot, Michael Neuder, Potuz, Justin Traglia, and Terence Tsao. EIP-7732: Enshrined Proposer-Builder Separation. Ethereum Improvement Proposal (Draft), 2024. Accessed: 2026-05-09. URL: https://eips.ethereum.org/EIPS/eip-7732.

11 Decentralized Systems Lab, Yale University. Mempool Guru, 2026. Accessed: 2026-05-09. URL: https://mempool.guru/.

12 Ethereum.org. Ethereum Consensus Layer Specifications, 2025. Accessed: 2026-05-09. URL: https://github.com/ethereum/consensus-specs.

13 Ethereum.org. Hegota Upgrade EIP Proposal Timelines, 2025. Accessed: 2026-05-09. URL: https://blog.ethereum.org/2025/12/22/hegota-timeline.

14 Ethereum.org. Staking as a Service, 2025. Accessed: 2026-05-09. URL: https://ethereum.org/en/staking/saas/.

15 Flashbots. MEV-Boost, 2022. Accessed: 2026-05-09. URL: https://github.com/flashbots/mev-boost.

16 Flashbots. Mempool Dumpster, 2025. Accessed: 2026-05-09. URL: https://github.com/flashbots/mempool-dumpster.

17 Foundry. Anvil – Ethereum Development Node. Accessed: 2026-05-09. URL: https://book.getfoundry.sh/anvil/.

18 Elijah Fox, Mallesh M. Pai, and Max Resnick. Censorship Resistance in On-Chain Auctions. In *5th Conference on Advances in Financial Technologies (AFT 2023)*. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPIcs.AFT.2023.19.

19 Pranav Garimidi, Joachim Neu, and Max Resnick. Multiple Concurrent Proposers: Why and How. 2025. arXiv:2509.23984.

20 Vincent Gramlich, Dennis Jelito, and Johannes Sedlmeir. Maximal Extractable Value: Current Understanding, Categorization, and Open Research Questions. *Electronic Markets*, page 49, 2024. doi:10.1007/s12525-024-00727-x.

21 Barbara Guidi and Andrea Michienzi. Linking MEV Attacks to Further Maximise Attackers' Gains: Evidence from the Ethereum Blockchain. *Blockchain: Research and Applications*, 2025. doi:10.1016/j.bcra.2025.100340.

22 Lioba Heimbach, Lucianna Kiffer, Christof Ferreira Torres, and Roger Wattenhofer. Ethereum's Proposer-Builder Separation: Promises and Realities. In *Proceedings of the 2023 ACM Internet Measurement Conference (IMC)*, pages 406–420. ACM, 2023. doi:10.1145/3618257.3624824.

23 Lioba Heimbach, Yann Vonlanthen, Juan Villacis, Lucianna Kiffer, and Roger Wattenhofer. Deanonymizing Ethereum Validators: The P2P Network Has a Privacy Issue. In *34th USENIX Security Symposium*, Seattle, WA, USA, 2025.

24 Mahimna Kelkar, Fan Zhang, Steven Goldfeder, and Ari Juels. Order-Fairness for Byzantine Consensus. In *Advances in Cryptology – CRYPTO 2020*, page 451–480. Springer-Verlag, 2020. doi:10.1007/978-3-030-56877-1_16.

25 Aya Miyaguchi and Vitalik Buterin. The Ethereum Foundation's Vision. Accessed: 2026-05-09. URL: https://blog.ethereum.org/2025/04/28/ef-vision.

26 Matthias Nadler, Katrin Schuler, and Fabian Schär. Blockchain Price Oracles: Accuracy and Violation Recovery. *Journal of Corporate Finance*, 2026. doi:10.1016/j.jcorpfin.2025.102908.

27 Michael Neuder and Terence Tsao. EIP-7547: Inclusion Lists. Ethereum Improvement Proposal (Stagnant), 2024. Accessed: 2026-05-09. URL: https://eips.ethereum.org/EIPS/eip-7547.

28 Paradigm. Reth: A Modular, Contributor-Friendly Implementation of the Ethereum Protocol, 2026. Accessed: 2026-05-09. URL: https://github.com/paradigmxyz/reth.

29 Fred B. Schneider. Implementing Fault-Tolerant Services Using the State Machine Approach: A Tutorial. *ACM Computing Surveys*, pages 299–319, 1990. doi:10.1145/98163.98167.

30 Caspar Schwarz-Schilling and Francesco D'Amato. Proposer Boost Considerations. Ethereum Notes, 2024. Accessed: 2026-05-09. URL: https://notes.ethereum.org/@casparschwa/H1T0k7b85.

31 Bence Soóki-Tóth, István András Seres, Kamilla Kara, Ábel Nagy, Balázs Pejó, and Gergely Biczók. Bribers, Bribers on The Chain, Is Resisting All in Vain? Trustless Consensus Manipulation Through Bribing Contracts. Cryptology ePrint Archive, Paper 2025/1719, 2025. URL: https://eprint.iacr.org/2025/1719.

32 Patrick Spiesberger, Nils Henrik Beyer, and Hannes Hartenstein. Why Ethereum Needs Fairness Mechanisms That Do Not Depend on Participant Altruism, 2026. arXiv:2603.05666.

33 Patrick Spiesberger, Jan Droll, and Hannes Hartenstein. A Policy-Based Mitigation of Transaction Exclusion in Ethereum (Work in Progress). In *Proceedings of the 30th ACM Symposium on Access Control Models and Technologies (SACMAT)*, pages 127–132, Stony Brook, NY, USA, 2025. ACM. doi:10.1145/3734436.3734458.

34 Thomas Thiery, Francesco D'Amato, Julian Ma, Barnabé Monnot, Terence Tsao, Jacob Kaufmann, and Jihoon Song. EIP-7805: Fork-Choice Enforced Inclusion Lists (FOCIL), 2024. Accessed: 2026-05-09. URL: https://eips.ethereum.org/EIPS/eip-7805.

35 Terence Tsao. Spec'ing Out Forward Inclusion-List with Dedicated Gas Limits, 2023. Accessed: 2026-05-09. URL: https://ethresear.ch/t/specing-out-forward-inclusion-list-w-dedicated-gas-limits/.

36 Sarisht Wadhwa, Julian Ma, Thomas Thiery, Barnabé Monnot, Luca Zanolini, Fan Zhang, and Kartik Nayak. AUCIL: An Inclusion List Design for Rational Parties. Cryptology ePrint Archive, Paper 2025/194, 2025. URL: https://eprint.iacr.org/2025/194.

37 Anton Wahrstätter, Jens Ernstberger, Aviv Yaish, Liyi Zhou, Kaihua Qin, Taro Tsuchiya, Sebastian Steinhorst, Davor Svetinovic, Nicolas Christin, Mikolaj Barczentewicz, and Arthur Gervais. Blockchain Censorship. In *Proceedings of the ACM Web Conference 2024 (WWW)*, pages 1632–1643, Singapore, 2024. ACM. doi:10.1145/3589334.3645431.

38 Gavin Wood. Ethereum: A Secure Decentralised Generalised Transaction Ledger. Ethereum Yellow Paper, Shanghai revision. Accessed: 2026-05-09, 2025. URL: `https://ethereum.github.io/yellowpaper/paper.pdf`.

39 Fei Wu, Thomas Thiery, Stefanos Leonardos, and Carmine Ventre. Strategic Bidding Wars in On-Chain Auctions. *IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*, 2024:503–511, 2024. `doi:10.1109/ICBC59979.2024.10634354`.

40 Sen Yang, Kartik Nayak, and Fan Zhang. Decentralization of Ethereum's Builder Market. In *Proceedings of the 2025 IEEE Symposium on Security and Privacy (SP)*, pages 1512–1530. IEEE, 2025. `doi:10.1109/SP61157.2025.00157`.

A Notation and Abbreviations

Symbol / Abbreviation	Meaning
Entities and sets	
V, V_s	validator set; slot-s committee
v_s^P	proposer of slot s ($v_s^P \in V_s$)
β_s	block assembler of slot s
B_s	block produced in slot s by β_s
state(B_s)	execution state after executing B_s
Pool_s^n	mempool view of node n at slot s
ExtendedPool_s^v	validator v's extended mempool (Algorithm 2, Check 3)
Transactions and fees	
τ	transaction
gas(τ)	gas consumption of τ
$f^\tau_{\text{tip, gas}}$	per-gas tip of τ, paid to block assembler β_s
$f^\tau_{\text{tip, tx}}$	per-transaction tip, $f^\tau_{\text{tip, tx}} := f^\tau_{\text{tip, gas}} \cdot \text{gas}(\tau)$
$f^\tau_{\text{base, gas}}$	per-gas base fee of τ (burned)
$f^\tau_{\text{base, tx}}$	per-transaction base fee, $f^\tau_{\text{base, tx}} := f^\tau_{\text{base, gas}} \cdot \text{gas}(\tau)$
c_{bribe}	on-chain bribing-contract overhead (cost)
ϵ	protocol deviation incentive (cost)
Block and reward quantities	
$g_{\max}(B_s)$	gas limit of block B_s
$\mathcal{R}(B_s)$	total block reward of B_s (transaction tips + MEV)
$\mathcal{R}_{\text{pub}}(B_s)$	tip revenue from public transactions in B_s
$\Delta(\tau, s)$	residual gas capacity above τ at slot s
Censorship (Resistance) notions	
$\mathcal{C}_\tau^{\text{Eth}}$	current per-slot cost of excluding τ
$\mathcal{A}$	adversary aiming to censor τ
$\mathcal{M}$	adversary's per-slot bribing budget
$(\mathcal{M}, \sigma)$-CR	censorship resistance notion (Definition 3)
Π_n	CR mechanism for n-slot censorship mitigation (Definition 5)
κ	committee size of a Π_n mechanism
$\mathcal{T}_{\text{req}}$	transactions required by Π_n in a specified block
FairFIL_s	inclusion list / FairFIL of slot s
FairBlock_s	local censorship-free block of a validator

Table 3 Symbols and abbreviations used throughout this paper.

B Mempool Consistency

Figure 3 reports per-node mempool consistency relative to the censorship-free blocks constructed in Section 5.1. To measure mempool consistency, we proceed as follows. For every censorship-free block, we consider the set of transactions contained therein and record the earliest first-observation timestamp of each transaction across all monitors. We then determine, for each monitor, the fraction of these transactions observed within 3 s of that timestamp. Each data point on the vertical axis therefore corresponds to a single censorship-free block and quantifies the share of the required transaction set that was timely available at the respective monitor. The horizontal axis distinguishes the three publicly available monitors of [11] in Dallas (US), Frankfurt am Main (GER), and Oregon (US). In the median, every monitor observes all transactions required to reconstruct a censorship-free block. Deviations occur but are confined to a small share of blocks and remain quantitatively small; we characterize their magnitude via the 1st and 2nd percentile of the per-node distribution. For instance, for Frankfurt am Main, P2 = 98.55 % indicates that in 98 % of the observed censorship-free blocks, no more than 1.45 % of required transactions were missing within the three-second time window. The lower P1 of 89.36 % is likely caused by a short-lived monitoring outage, as the missed transactions cluster across several consecutive blocks. Dallas and Oregon exhibit the same pattern at slightly higher percentile values.

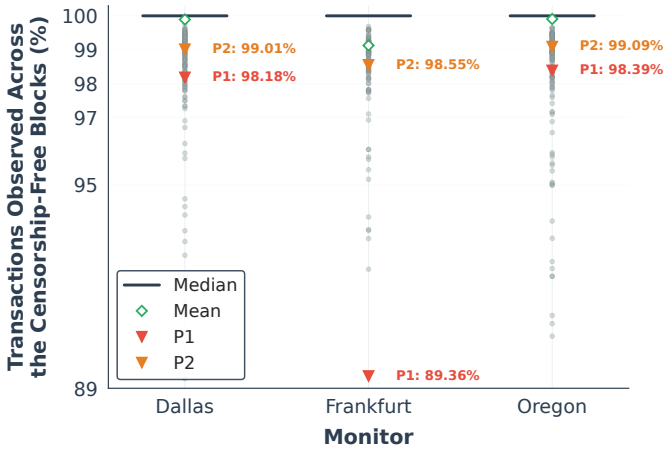

Figure 3 Per-node mempool consistency during September 2025 at the three monitors of [11]. The vertical axis shows, per censorship-free block, the fraction of required transactions observed within 3 s of their earliest first-observation across all monitors.

C Proof of Lemma 4 (Ethereum baseline)

Proof. Let $\mathcal{M} := f^{\tau}_{\text{tip, tx}} + c_{\text{bribe}}$. Lemma 4 consists of two claims, which we prove in turn. First, we show that an adversary with per-slot budget $\mathcal{M} + \epsilon$ can exclude transaction τ from every block, establishing the per-slot exclusion cost $\mathcal{C}^{\text{Eth}}_{\tau}$. Second, we show that no adversary with per-slot budget $\mathcal{M}$ can delay τ even by a single slot, establishing $(\mathcal{M}, 0)$-censorship resistance.

Per-slot budget $\mathcal{M}+\epsilon$ suffices to censor τ in every slot. Adversary $\mathcal{A}$ bribes the block assembler β_s through the bribing contract [31] to exclude τ from block B_s. Of the budget $\mathcal{M}+\epsilon = f^{\tau}_{\text{tip, tx}} + c_{\text{bribe}} + \epsilon$, the contract consumes the overhead c_{bribe}, leaving a bribe of $f^{\tau}_{\text{tip, tx}} + \epsilon$ paid to β_s upon observed exclusion of τ from B_s. By including τ, β_s would earn the per-transaction tip $f^{\tau}_{\text{tip, tx}}$; by excluding τ and accepting the bribe, β_s earns $f^{\tau}_{\text{tip, tx}} + \epsilon$, which strictly exceeds the inclusion payoff for any $\epsilon > 0$. A rational block assembler therefore excludes τ. $\mathcal{A}$ can repeat this attack in every slot at the same per-slot cost $\mathcal{C}^{\text{Eth}}_{\tau} = f^{\tau}_{\text{tip, tx}} + c_{\text{bribe}} + \epsilon$, so τ is never included.

Budget $\mathcal{M}$ does not suffice to delay τ even by a single slot. Assume, for contradiction, that an adversary with per-slot budget $\mathcal{M}$ can cause τ to be excluded from B_s. Then the bribe paid to β_s is at most $\mathcal{M} - c_{\text{bribe}} = f^{\tau}_{\text{tip, tx}}$. Accepting this bribe yields β_s exactly the same payoff as including τ and earning the tip $f^{\tau}_{\text{tip, tx}}$. By Section 2.2, in case of indifference, a rational β_s follows the protocol and includes τ in B_s, contradicting the assumed exclusion. Hence no adversary with per-slot budget $\mathcal{M}$ can delay τ for any slot. Therefore, Ethereum in its current state is $(\mathcal{M}, 0)$-censorship-resistant. ◀

D Proof of Lemma 12 (Protocol-deviation cost)

Proof. Let $\mathcal{M} := \min\big(\mathcal{R}(B_s), \mathcal{R}(B_{s+1})\big) + c_{\text{bribe}}$ and, without loss of generality, assume $\mathcal{R}(B_{s+1}) \leq \mathcal{R}(B_s)$ so that $\mathcal{M} = \mathcal{R}(B_{s+1}) + c_{\text{bribe}}$. We further assume the condition of Lemma 12: two-slot censorship of τ requires a deviation from FairFIL (i.e., suppression is infeasible within the considered budget). We further assume for all proofs that $\mathcal{C}^{\text{Eth}}_{\tau} \leq \mathcal{M}$.

Enumeration of deviation strategies. For two-slot censorship, transaction τ must be censored from both B_s and B_{s+1}. Under the assumption, this requires one of the following protocol deviations:

Strategy (a): Construction failure. Adversary $\mathcal{A}$ bribes block assembler β_s to construct a non-compliant FairFIL$_s$ that omits τ (Property 2). A majority of V_s rejects B_s (Lemma 9); β_s forfeits the full block reward $\mathcal{R}(B_s)$.

Strategy (b): Enforcement failure. Adversary $\mathcal{A}$ bribes β_{s+1} to omit $\tau \in$ FairFIL$_s$ from B_{s+1} (Property 4). A majority of V_{s+1} rejects B_{s+1} (Lemma 11); β_{s+1} forfeits $\mathcal{R}(B_{s+1})$.

Budget $\mathcal{M}+\epsilon$ suffices to censor τ for two slots. With $\mathcal{R}(B_{s+1}) \leq \mathcal{R}(B_s)$, adversary $\mathcal{A}$ uses Strategy (b):

1. In slot s, $\mathcal{A}$ bribes β_s at the Ethereum baseline cost $\mathcal{C}^{\text{Eth}}_{\tau}$ (Lemma 4) to exclude τ from B_s. As τ remains includable in B_{s+1} under Fair Ordering, τ enters FairFIL$_s$ by Lemma 9. The slot-s expenditure is $\mathcal{C}^{\text{Eth}}_{\tau} \leq \mathcal{M}$.
2. In slot $s+1$, $\mathcal{A}$ bribes β_{s+1} through the bribing contract to omit $\tau \in$ FairFIL$_s$ from B_{s+1}, resulting in a loss of block reward $\mathcal{R}(B_{s+1})$. The bribe must compensate the forfeiture of $\mathcal{R}(B_{s+1})$ and strictly exceed it by some $\epsilon > 0$ to make omission strictly preferable for a rational β_{s+1} (Section 2.2); the contract additionally consumes the overhead c_{bribe}. The slot-s+1 expenditure is therefore $\mathcal{R}(B_{s+1}) + c_{\text{bribe}} + \epsilon = \mathcal{M} + \epsilon$.

The per-slot peak expenditure for two-slot censorship is $\mathcal{M}+\epsilon$. By Property 4, B_{s+1} is rejected, and no obligations apply to the subsequent block assembler β_{s+2}. The resulting delay is $\sigma \geq 2$. Thus, FairFIL is at most $(\mathcal{M}, 1)$-censorship-resistant. The case $\mathcal{R}(B_s) \leq \mathcal{R}(B_{s+1})$ is argumentatively identical, requiring an attack via Strategy (a) and the payment of the baseline cost in slot $s+1$.

Budget $\mathcal{M}$ does not suffice to censor τ for two slots. Assume adversary $\mathcal{A}$ has per-slot budget $\mathcal{M}$ and attempts Strategy (b). Of the budget, the bribing contract consumes the overhead c_{bribe}, leaving at most $\mathcal{R}(B_{s+1})$ payable to β_{s+1}. By accepting, β_{s+1} forfeits $\mathcal{R}(B_{s+1})$, yielding the same net payoff as honest compliance. By the rationality model of Section 2.2, in case of indifference β_{s+1} follows the protocol and includes τ in B_{s+1}. For Strategy (a), the analogous bribe to β_s falls strictly short of the higher loss $\mathcal{R}(B_s) \geq \mathcal{R}(B_{s+1})$, so β_s strictly prefers compliance. Hence no deviation occurs within budget $\mathcal{M}$, and by the assumption, two-slot censorship is infeasible. Therefore, FairFIL is $(\mathcal{M}, 1)$-censorship-resistant. ◀

E Proof of Lemma 13 (Suppression cost)

Proof. Let $\mathcal{M} := f^*(\tau) \cdot \Delta(\tau, s)$. We assume that under Ethereum's transaction fee mechanism [8], blocks are typically not fully utilized – by design, the average target fullness is approximately half of the gas limit [8]. Consequently, an adversary populating residual gas capacity above τ in Fair Ordering must contribute gas that would otherwise remain unused, so $\Delta(\tau, s)$ is large relative to the gas of a single transaction. Based on our observations in Section 5.1, we therefore assume throughout that $\mathcal{C}_\tau^{\text{Eth}} < f^*(\tau) \cdot \Delta(\tau, s) = \mathcal{M}$.

Budget $\mathcal{M} + \epsilon$ suffices to censor τ for two slots. Adversary $\mathcal{A}$ proceeds as follows: $\mathcal{A}$ propagates a filler transaction τ' through the P2P layer such that $\tau' \succ \tau$ under Fair Ordering and the gas usage of τ' strictly exceeds $\Delta(\tau, s)$. The parameter ϵ is required to cover the residual gas cost needed to strictly exceed $\Delta(\tau, s)$. We simplify the block capacity optimization problem and assume that ϵ is sufficiently large such that replacing τ' with a lower-tip transaction would not increase total revenue through higher gas usage. Accordingly, the rational block assembler β_s includes τ' in B_s to capture its tip, yielding an adversarial expenditure strictly greater than $f^*(\tau) \cdot \Delta(\tau, s) = \mathcal{M}$, where $f^*(\tau) = f^{\tau}_{\text{tip, gas}} + f^{\tau'}_{\text{base, gas}}$ is the per-gas cost required to outrank τ. Including τ' in B_s exhausts the available gas capacity above τ in the Fair Ordering: τ is no longer includable in B_s, and by Definition 1, $\tau \notin \text{FairFIL}_s$. Since $\tau \notin \text{FairFIL}_s$, block assembler β_{s+1} has no obligation to include τ in B_{s+1}. The baseline bribe in slot $s+1$ is therefore sufficient to ensure that τ is also not included in B_{s+1}. Since $\mathcal{C}_\tau^{\text{Eth}} < \mathcal{M} + \epsilon$, this per-slot budget suffices for adversary $\mathcal{A}$ to censor τ for two slots.

Budget $\mathcal{M}$ does not suffice for two-slot censorship via suppression. Assume, for contradiction, that an adversary with budget $\mathcal{M}$ achieves two-slot censorship solely via the transaction suppression attack. For $\tau \notin B_{s+1}$ to hold without violating the enforcement of FairFIL, we require $\tau \notin \text{FairFIL}_s$; otherwise Lemma 11 implies $\tau \in B_{s+1}$. For $\tau \notin \text{FairFIL}_s$, the cumulative gas of transactions ranking strictly above τ in the Fair Ordering must exceed $\Delta(\tau, s)$, i.e., the remaining gas capacity after including τ under Fair Ordering in the block. With budget $\mathcal{M} = f^*(\tau) \cdot \Delta(\tau, s)$, the adversary can exactly fill the available gas capacity above τ in B_s, making τ just feasible for inclusion. By the definition of $\Delta(\tau, s)$, this already exhausts transactions ranking above τ, implying that τ becomes the next transaction to be considered under Fair Ordering. Consequently, given budget $\mathcal{M}$, τ is either included directly in B_s or, by Lemmas 8 and 9, appears in FairFIL_s and is therefore included in B_{s+1}. Accordingly, given budget $\mathcal{M}$, adversary $\mathcal{A}$ cannot use the transaction suppression attack to achieve two-slot censorship, contradicting the assumption. ◀

F Censorship under Prior Censorship Resistance Mechanisms

The following four lemmas bound the censorship resistance of FIL, FOCIL, MCP, and AUCIL, surveyed in Section 3. The proofs share a common structure: for each mechanism, we describe an adversary $\mathcal{A}$ with the per-slot bribing budget $\mathcal{M}$ that prevents the inclusion of an arbitrary target transaction τ for an unbounded number of consecutive slots. By Definition 3, this establishes that the mechanism is not $(\mathcal{M}, 0)$-censorship-resistant for the selected bribing budget. We do not claim the stated budgets are tight; lower budgets may suffice.

F.1 Forward Inclusion Lists (FIL)

▶ **Lemma 14.** *With FIL [27], an adversary with per-slot budget $\mathcal{M} = \mathcal{C}_\tau^{\mathrm{Eth}} + \epsilon$ prevents the inclusion of transaction τ for an unbounded number of consecutive slots.*

Proof. We exhibit adversary $\mathcal{A}_{\mathrm{FIL}}$ with per-slot budget $\mathcal{M} = \mathcal{C}_\tau^{\mathrm{Eth}} + \epsilon$. In each slot s, $\mathcal{A}_{\mathrm{FIL}}$ bribes the block assembler β_s through a single bribing-contract call to (i) exclude τ from B_s and (ii) omit τ from $\mathcal{T}_{\mathrm{req}}$. For (i), Lemma 4 establishes that $\mathcal{C}_\tau^{\mathrm{Eth}} = f^\tau_{\mathrm{tip,\,tx}} + c_{\mathrm{bribe}} + \epsilon$ suffices to make a rational β_s exclude τ from B_s. For (ii), FIL provides no per-transaction reward for listing in $\mathcal{T}_{\mathrm{req}}$ [27], so β_s is indifferent between listing and omitting τ; an additional margin of ϵ makes omission strictly preferable. Since both deviations are settled through the same bribing-contract call, only one overhead c_{bribe} applies, and the total per-slot cost amounts to $\mathcal{M} = f^\tau_{\mathrm{tip,\,tx}} + c_{\mathrm{bribe}} + 2\epsilon = \mathcal{C}_\tau^{\mathrm{Eth}} + \epsilon$. After the attack, $\tau \notin B_s$ and $\tau \notin \mathcal{T}_{\mathrm{req}}$, so no obligation to include τ in block B_{s+1} arises. Hence, $\mathcal{A}_{\mathrm{FIL}}$ can repeat the attack in every subsequent slot at the same per-slot cost, and τ is never included. ◀

F.2 Fork-Choice Enforced Inclusion Lists (FOCIL)

▶ **Lemma 15.** *With FOCIL [34], an adversary with per-slot budget $\mathcal{M} = \kappa\,(\epsilon + c_{\mathrm{bribe}}) + \mathcal{C}_\tau^{\mathrm{Eth}}$ prevents the inclusion of transaction τ for an unbounded number of consecutive slots.*

Proof. We exhibit adversary $\mathcal{A}_{\mathrm{FOCIL}}$ with per-slot budget $\mathcal{M} = \kappa\,(\epsilon + c_{\mathrm{bribe}}) + \mathcal{C}_\tau^{\mathrm{Eth}}$. In each slot s, $\mathcal{A}_{\mathrm{FOCIL}}$ targets the inclusion committee V_s^{FOCIL} of κ includers and the block assembler β_s. For each of the κ includers, $\mathcal{A}_{\mathrm{FOCIL}}$ pays a bribe of ϵ together with the contract overhead c_{bribe} to omit τ from the includer's partial inclusion list. Since FOCIL provides no per-transaction reward to includers [34], the includer is indifferent between listing and omitting τ, and the margin ϵ tips the balance toward omission. The total committee cost is $\kappa\,(\epsilon + c_{\mathrm{bribe}})$. Once τ is omitted from all κ partial lists, the aggregate $\mathcal{T}_{\mathrm{req}}$ does not contain τ, and β_s is not obligated to include τ in B_s. However, a rational block assembler β_s would still include τ voluntarily to capture the transaction's tips $f^\tau_{\mathrm{tip,\,tx}}$. $\mathcal{A}_{\mathrm{FOCIL}}$ therefore additionally bribes β_s at the Ethereum baseline cost $\mathcal{C}_\tau^{\mathrm{Eth}}$ (Lemma 4) to exclude τ from B_s, yielding the stated total per-slot cost $\mathcal{M}$. Since FOCIL imposes no obligation on subsequent slots, $\mathcal{A}_{\mathrm{FOCIL}}$ can repeat the attack against the subsequently selected committee in every subsequent slot at the same per-slot cost, so τ is never included. ◀

F.3 Multiple Concurrent Proposers (MCP)

▶ **Lemma 16.** *With MCP [19], an adversary with per-slot budget $\mathcal{M} = \kappa \cdot \mathcal{C}_\tau^{\mathrm{Eth}}$ prevents the inclusion of transaction τ for an unbounded number of consecutive slots.*

Proof. We exhibit adversary $\mathcal{A}_{\mathrm{MCP}}$ with per-slot budget $\mathcal{M} = \kappa \cdot \mathcal{C}_\tau^{\mathrm{Eth}}$. MCP replaces the single block assembler with κ concurrent proposers, each producing an independent partial

block. $\mathcal{A}_{\text{MCP}}$ targets each of these κ proposers individually, considering the worst case in which every concurrent proposer attempts to include τ in its partial block. Including τ yields the per-transaction tip $f^{\tau}_{\text{tip, tx}}$ to the proposer [19]. $\mathcal{A}_{\text{MCP}}$ compensates each proposer for the forgone tip $f^{\tau}_{\text{tip, tx}}$, adds the margin ϵ to make exclusion strictly preferable, and pays the contract overhead c_{bribe}, totaling $\mathcal{C}^{\text{Eth}}_{\tau} = f^{\tau}_{\text{tip, tx}} + \epsilon + c_{\text{bribe}}$ per proposer. Across all κ proposers, the per-slot cost amounts to $\mathcal{M} = \kappa \cdot \mathcal{C}^{\text{Eth}}_{\tau}$. Since MCP imposes no obligation propagating beyond the current slot, $\mathcal{A}_{\text{MCP}}$ can repeat the attack against the subsequently selected set of κ proposers in every subsequent slot at the same per-slot cost, so τ is never included. ◀

F.4 Auction-Based Inclusion Lists (AUCIL)

▶ **Lemma 17.** *With AUCIL [36], an adversary with per-slot budget $\mathcal{M} = (\kappa + 1) \cdot \mathcal{C}^{\text{Eth}}_{\tau}$ prevents the inclusion of transaction τ for an unbounded number of consecutive slots.*

Proof. We exhibit adversary $\mathcal{A}_{\text{AUCIL}}$ with per-slot budget $\mathcal{M} = (\kappa + 1) \cdot \mathcal{C}^{\text{Eth}}_{\tau}$. AUCIL combines a constructing committee of κ includers with a separate block assembler β_s that produces B_s in the same slot, and rewards each includer for the transactions it contributes [36]. $\mathcal{A}_{\text{AUCIL}}$ adopts the strategy of $\mathcal{A}_{\text{MCP}}$ (Lemma 16) against the κ committee members: each includer is bribed at cost $\mathcal{C}^{\text{Eth}}_{\tau}$ to omit τ from its partial inclusion list, totaling $\kappa \cdot \mathcal{C}^{\text{Eth}}_{\tau}$. Once τ is omitted from all κ partial lists, the aggregate $\mathcal{T}_{\text{req}}$ does not contain τ, removing the protocol obligation on β_s. Unlike MCP, however, AUCIL retains a separate block assembler that produces B_s alongside the committee, and a rational β_s would still include τ to capture $f^{\tau}_{\text{tip, tx}}$. $\mathcal{A}_{\text{AUCIL}}$ therefore additionally bribes β_s at the Ethereum baseline cost $\mathcal{C}^{\text{Eth}}_{\tau}$ (Lemma 4) to exclude τ from B_s, yielding the stated total per-slot cost $\mathcal{M} = \kappa \cdot \mathcal{C}^{\text{Eth}}_{\tau} + \mathcal{C}^{\text{Eth}}_{\tau} = (\kappa + 1)\, \mathcal{C}^{\text{Eth}}_{\tau}$. Wadhwa et al. [36] provide a tighter bound on the censorship resistance of AUCIL. Since AUCIL imposes no obligation propagating beyond the current slot, $\mathcal{A}_{\text{AUCIL}}$ can repeat the attack against the subsequently selected committee and assembler in every subsequent slot at the same per-slot cost, so τ is never included. ◀