

RANDAO Manipulation in the Presence of MEV

Kaya Alpturer  

Princeton University, NJ, USA

Nicholas Hope  

Talarion, San Francisco, CA, USA

S. Matthew Weinberg  

Princeton University, NJ, USA

Abstract

Ethereum’s randomness beacon (RANDAO) is well-known to be manipulable, and prior work [2] computes the precise fraction of blocks a strategic proposer can propose. The fraction of blocks proposed, however, is only a proxy for participants’ rewards. We propose a generalized reward model capturing many canonical forms of rewards: consensus reward rollover, multi-block MEV, CEX-DEX arbitrage, oracle manipulation, and others. We provide a methodology that computes an ε -optimal strategy for any reward scheme in our model (and in particular, any combination of the above rewards). Finally, we apply our methodology to several canonical examples, and establish the sensitivity of RANDAO manipulation to the underlying rewards. We find that if rewards partially roll over, or scale super-linearly with consecutive blocks, the incentive to manipulate RANDAO is amplified. Lastly, we investigate tail-slot slashing, which can be modeled as a reward function, and show that honest equilibria can be recovered.

2012 ACM Subject Classification Theory of computation $\rightarrow$ Algorithmic game theory and mechanism design; Information systems $\rightarrow$ Digital cash; Security and privacy $\rightarrow$ Distributed systems security

Keywords and phrases Proof of Stake, Consensus, Blockchain, Ethereum, Randomness manipulation

Digital Object Identifier 10.4230/LIPIcs.AFT.2026.23

Supplementary Material *Software:* <https://github.com/kalpturer/randao-mev>
archived at `swh:1:dir:5fc25f28e664c64535e1f20e516a4f7ca0e47e76`

Funding This work was supported by an Ethereum Foundation grant.

Acknowledgements During Prof. Weinberg’s development of this paper, he participated as an expert witness on behalf of the State of Texas in ongoing litigation against Google (the “Google Litigation”).

1 Introduction

A major concern for blockchain protocols is the potential for economically rational participants to deviate from the protocol specification to maximize their rewards. In many protocols, randomness is used to ensure the fair distribution of block proposing rights. Thus, if the randomness in a protocol is not robust, deviations that bias it will be incentivized. In Bitcoin, protocol deviations such as selfish mining have relied mainly on forking and hiding blocks, as biasing the randomness requires breaking the robust SHA256 hash function. In Proof-of-Stake protocols, however, the randomness is often provided by a distributed randomness beacon built into the protocol itself, creating a novel attack vector. In particular, Ethereum’s implementation of randomness, called RANDAO, has been well known to be vulnerable to block withholding attacks [2]. Given RANDAO’s role in determining the winners of the block proposal lottery, there is great financial incentive to manipulate it.



© Kaya Alpturer, Nicholas Hope, and S. Matthew Weinberg;
licensed under Creative Commons License CC-BY 4.0

8th Conference on Advances in Financial Technologies (AFT 2026).

Editors: Aggelos Kiayias and Maria Kyropoulou; Article No. 23; pp. 23:1–23:22

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

With Ethereum’s stake pool at roughly \$80 billion, the concern for RANDAO manipulation has resulted in several lines of work, analyzing the potential impacts of this type of deviation [2, 20, 27, 28]. These works establish that an adversary with an α fraction of the total stake is able to propose strictly more than an α fraction of the total blocks, and therefore earn greater than an α fraction of the total consensus rewards.

However, rewards in Ethereum have increasingly been dominated by execution layer rewards that stem from decentralized finance activity as opposed to constant rewards per block proposed. We have seen, for example, that sophisticated actors are able to extract rewards from arbitrage, front-running, and transaction fees [11, 21]. Therefore, there is a need to model rewards more generally to include higher-order reward properties beyond a single block.

The importance of varying rewards is well-established in the proof-of-work setting. For Bitcoin, [10] shows that variance in transaction fees fundamentally alters the equilibrium structure of selfish mining, with subsequent work generalizing the analyses to richer reward models [6, 7]. For RANDAO manipulation, however, the analogous question of how MEV and similar non-uniform reward structures affect the incentive to manipulate it has remained open. While prior work laid the foundation for the MDP-based analysis and formalization of the larger problem, these analyses mainly measure the fraction of blocks proposed.

Our primary contribution is a sampling-based methodology to compute ϵ -optimal policies for RANDAO manipulation for significantly more general reward models. Our approach relies on a novel truncation of the RANDAO game that can provably provide bounds on the optimal solution. Prior work [2] relies critically on uniform rewards to collapse the state space, and our truncation and sampling-based approach requires a new argument for bounding the optimal manipulation.

As a warmup exercise, we show that our methodology reproduces the results of [2] in the case of uniform rewards (single unit of reward per proposed block). Our approximate solver is able to more efficiently compute approximate results, and larger runs with sufficient samples are capable of achieving progressively more accurate results. We then investigate several reward models relevant to transaction fees, arbitrage, oracle manipulation, and consensus rewards. In the process, we characterize specific types of MEV and their interactions with RANDAO manipulation, including *per-block rewards (with partial rollover)*, and two models of *multi-block MEV*.

1.1 Overview of Methodology

Below, we overview several instances of execution layer rewards, and how they fit into our methodology. Recall that RANDAO proceeds as follows. Every time a proposer proposes a block during their slot, RANDAO changes, becoming a new value that is: (a) deterministic, to the proposer, (b) appears uniformly random, to anyone without the proposer’s private key. At the end of every 32-block epoch, the next 32 proposers are selected according to the current RANDAO value.¹ Therefore, when proposers are selected for the last k blocks of an epoch, they can selectively propose/skip slots to select among 2^k independent random options for the next epoch’s 32 proposers. The cost of doing so is whatever rewards are sacrificed from skipped blocks. The benefit of doing so is the potential for more rewards (and perhaps more opportunities for future manipulation) in the next epoch. We call the largest such k controlled by an adversary the *tail*.

¹ In the actual Ethereum implementation, the RANDAO from epoch i determines the proposers for epoch $i + 2$. We ignore interleaving and discuss its effect in the model assumptions.

When discussing sample rewards below, we use `TAIL` to refer to the pattern of proposed/skipped slots by the adversary in the tail of an epoch and `NEXT` for the pattern of proposed slots by the adversary in the non-tail. Assuming that other proposers honestly follow the intended protocol (and propose in every slot), a strategic proposer who does not control the last slot of an epoch optimizes their expected rewards by proposing in every non-tail slot. This is simply because: (a) such a proposer has no control over the next epoch's RANDAO, (b) a proposer seeking to optimize rewards in this epoch alone can always replace a skipped block with an empty block without affecting the execution rewards and still collect block rewards.

► **Example 1 (Reward Rollover).** Many execution layer rewards, such as those from on-chain arbitrage and sandwich attacks, roll over if not claimed in a block. For instance, an MEV opportunity in the mempool simply becomes available to the next proposer after a missed block.² This also occurs with consensus layer rewards. The block reward increases after a miss due to the increase in the number of attestations that can be included (see Figure 6). Intuitively, this rollover helps RANDAO manipulation by reducing the cost of missing a block if the adversary knows that they will get to propose after a miss. Because this rollover can be represented as a function on `TAIL` and `NEXT`, we investigate its effect in our per-slot reward model.

► **Example 2 (Arbitrage Value).** For an adversary controlling k consecutive blocks, a single arbitrage at the end is more profitable than multiple arbitrage transactions due to trading fees [19]. Fees introduce a profitability threshold, giving the arbitrage an option-like payoff.³ Controlling a longer sequence of k blocks increases the probability that the price will drift beyond this threshold. Thus, a validator intending to perform CEX-DEX arbitrage values a run of k consecutive blocks more than k independent blocks, motivating a superlinear model for MEV extraction. In particular, observe again that the reward earned by the proposer can be written as a function on `TAIL` and `NEXT`.

► **Example 3 (Oracle Manipulation).** Manipulation of time-weighted average price (TWAP) oracles in order to force illegitimate liquidations is a well-understood form of multi-block MEV [18]. While oracle manipulation of this form is already a concern without an adversary controlling multiple blocks in a row⁴, RANDAO manipulation exacerbates the threat by allowing an adversary complete control over transaction inclusion, reducing the attack cost. For a given adversary, TWAP attacks represent a model in which a large, fixed value (the liquidation profit) is available if the validator can secure enough sequential blocks to initiate the liquidation. As such rewards can be written as a function on `TAIL` and `NEXT`, we explore this further in the threshold multi-block MEV model.

² See Figure 7 for the distribution of MEV bids on the MEV-Boost auction showing the distribution of these rewards.

³ For instance, consider a scenario in which DEX and CEX prices begin perfectly equal, with a de minimis fee parameter on the CEX but a 30 basis point fee on the DEX. In such a scenario, a price discrepancy of more than 30 basis points must arise via price movement on the CEX before a validator may execute an arbitrage transaction profitably. Reserving sole arbitrage rights via multi-slot control thus exhibits the same nonlinear payoff function as a put or call option: the opportunity is worthless until the option becomes in the money (that is, the price discrepancy exceeds cumulative fee parameters of the target CEX-DEX pair).

⁴ Prior work [18] primarily considers block proposers that include the attacker's transactions, but also transactions from arbitrageurs that de-manipulate asset prices. Such a model remains appropriate in a setting where the sequence of proposers cannot be guaranteed in advance, such as Proof-of-Work in contrast to Ethereum.

Our Methodology. All of the above examples can be captured by specifying two functions $f(\cdot)$ and $g(\cdot)$. $f(\cdot)$ takes as input a sequence of proposals by the adversary in the non-tail part of the epoch (which we refer to as NEXT), and outputs “under the specified reward model, what is the expected reward for the adversary if the adversary proposes all of their blocks and honest validators propose the other blocks?” $g(\cdot)$ takes as input a fully specified TAIL sequence and outputs “under the specified reward model, what is the expected reward for the adversary if the adversary proposes the specified blocks and misses the others in TAIL?” Note the distinction between the two: in NEXT the non-proposed blocks are proposed by an honest party and in TAIL non-proposed blocks are missed. As we later explore, this may make a difference if honestly proposed blocks affect rewards of other blocks. For *any* reward model that can be fully specified by some $f(\cdot)$ and $g(\cdot)$ (including all three examples above, and any combination of them), our methodology computes an ϵ -optimal strategy for a strategic proposer. In the formal model of Section 2, f corresponds to $\text{next}(\cdot)$ and g corresponds to $\text{propose}(\cdot)$.

Highlights. In addition to developing our methodology, we also apply it to explore canonical settings and draw conclusions about the role of execution rewards in RANDAO incentives. First, we reproduce the results of [2] in the case of uniform rewards (constant per block), and show that our approximate solver can quickly reproduce these results at coarse ϵ . We then demonstrate the methodology on three execution-layer reward models: per-slot rollover, a smooth multi-block model, and a threshold multi-block model. Consensus-reward rollover is directly visible in the data (Figure 6), but MEV rollover is harder to verify empirically: slot misses are rare on Ethereum, so the available observational data on what happens to MEV opportunities across a missed slot is scarce. Systematic multi-block MEV is also weak in current Ethereum [26], so the smooth and threshold models are stylized. The modelled MEV regimes may matter in the future under block-building redesigns. In the regimes we model, partial reward rollover and multi-block MEV both amplify the incentive to manipulate RANDAO.

Lastly, we focus on two practically relevant investigations using the same framework. We show how increasing the epoch length decreases the manipulation gains across reward models⁵. We then encode, as a reward function, a tail-slashing curve that penalizes misses toward the tail more heavily and amplifies penalties for groups of consecutive misses, and show that honest behavior can be made ϵ -optimal under appropriate slashing magnitudes.

1.2 Related Work

RANDAO Manipulation. Since its inception, RANDAO has been known to be manipulable to a certain degree by withholding tail slots [13, 28]. Early probabilistic modeling based work [3] shows that this manipulation is small but non-negligible. [2] develops a Markov Decision Process (MDP) formulation of RANDAO manipulation to compute optimal strategies. Later work extends this analysis to consider forking [20, 27]. Aside from RANDAO, prior work also considers manipulation of other proof-of-stake protocols [9, 15, 16]. However, all prior work considers strategic proposers who seek only to optimize their fraction of proposed slots, and do not consider variable execution rewards.

⁵ [2] observed this for uniform rewards. Our results extend it to non-uniform reward structures, strengthening the case for longer epochs as a mitigation.

Selfish Mining. The analogous manipulation in Bitcoin is selfish mining, first analyzed by [14]. Subsequent work improves the analysis and considers optimal strategies [7, 23, 30] via an MDP methodology. Several works consider the effect of transaction fees on selfish mining. [10] shows that transaction fees radically change the analysis, and selfish mining variants can be profitable for arbitrarily small miners [7]. In a similar spirit to our work, [6] develops a generalized model of selfish mining to accommodate execution layer rewards. In comparison to these works, ours is the first to consider execution rewards such as MEV extracted through multiple consecutive blocks. However, our model does not subsume these – a strategic proposer in [6, 10] can adapt their strategy on the realized draw of transaction fees, whereas a proposer in our model does not. Moreover, our methodology finds the optimal strategy, rather than analyzing specific strategies that outperform honesty.

Computing Optimal Manipulations. [23] are the first to use MDP methodology to compute optimal manipulations in a blockchain protocol. MDPs are also used in [2, 7, 30], and similar convergence-based techniques are used in [16]. In comparison to these works, ours is the first to find optimal strategies for execution rewards beyond the uniform block rewards. In comparison to [2] specifically, our methodology can estimate ϵ -optimal strategies more efficiently and accommodate significantly more complex reward structures via sampling.

Maximal Extractable Value. Maximal extractable value (MEV) has been a large and growing part of reward modeling [4, 11]. Recently, there has been interest in understanding the game-theoretic implications of MEV on the incentive-compatibility of blockchain protocols [5, 17, 25]. MEV’s interaction with various kinds of protocol deviations is an active area of research.

Validator Rewards and Slashing. Modifying validator rewards to disincentivize strategic deviations is a recurring theme in blockchain protocol design. Ethereum currently penalizes missed attestations uniformly via the inactivity leak [13], and there have been community discussions of introducing missed-slot penalties as a more targeted intervention [22]. A parallel line of work proposes time-sensitive or retroactive block rewards to disincentivize strategic delays in block proposal (timing games) [1, 24, 25]. We investigate a similar mitigation for RANDAO manipulation: we model tail slashing penalties as a reward function within our framework, and apply our methodology to determine slot miss penalties at the tail under which honest behavior is ϵ -optimal.

1.3 Main Contributions

- We analyze RANDAO manipulation under a wide array of stylized reward models (Section 4). This improves prior work by: (a) being the first to consider varying rewards such as execution layer rewards, and (b) allowing quick exploratory runs at coarse ϵ .
- We introduce a novel truncation of the RANDAO game that provides provable upper and lower bounds (Section 3), rendering many instances of the game with complicated reward functions computationally tractable.
- We prove robustness guarantees for our methodology, and provide a codebase⁶ for easy replication of our results and running similar analyses for any reward function of interest.

⁶ <https://github.com/kalpturer/randao-mev>.

- We apply our methodology to canonical settings of interest (Section 5, Figure 2), and conclude that the effect of RANDAO manipulation is highly dependent on the underlying reward model.
- We apply our framework to evaluate practical mitigations: epoch-length variation (Figure 4) and tail slashing penalties (Section 6). Modeling slashing as a reward function within our methodology, we compute, for each MEV reward model, minimal slashing curves that recover honest behavior at the equilibrium.

1.4 Roadmap

In Section 2, we model RANDAO manipulation with general rewards. In Section 3, we present our methodology of computing ε -optimal policies. We then proceed to analyze complex reward models in Section 4. In Section 5, we compare different reward models and show how much better a strategic proposer can do compared to honest proposers, including a sensitivity analysis on the epoch length. In Section 6, we analyze slashing-based mitigation approaches under which honest behavior is ε -optimal. Finally, Section 7 concludes with a summary and future work. We defer proofs to the appendix.

2 Modeling RANDAO

We now model RANDAO manipulation in its full generality, where the reward function is dynamic.

2.1 Background

Ethereum Consensus. Ethereum uses a Proof-of-Stake consensus mechanism which divides time into epochs with 32 slots, each of which lasts for 12 seconds. The participants in the protocol are called *validators* and stake ≥ 32 ETH each. For each epoch, 32 validators are sampled uniformly at random to propose the blocks in the epoch. The sampling procedure requires a reliable source of randomness, which is provided by the RANDAO mechanism [13].

RANDAO Mixing. Each block contains a 256-bit random seed called the RANDAO. The validator deterministically computes the block RANDAO by signing the current epoch number and then hashing the resulting signature. This hash is then mixed into the RANDAO value via an XOR operation. Note that this mixing is verifiable since the public key of the validator is known and the validator also includes the raw signature in the block. Thus, there is a unique value which a validator may use when updating the RANDAO. The future value of the RANDAO nonetheless cannot be computed by any third party ahead of time, since it relies upon both (a) the validator's private key, and (b) the RANDAO value as of the preceding block. The only exception to this rule occurs if and when a sequence of validators collude to reveal RANDAO values in advance. We model such a group of validators as the adversarial proposer.

Proposer Selection. The RANDAO value at the end of epoch E is used to determine the proposers for the next epoch by deterministically sampling 32 validators. The sampling protocol is complex, but can be accurately modeled by 32 independently and uniformly random selections from the pool of available validators.

Rewards. Ethereum employs a reward mechanism that incentivizes validators to propose, vote on blocks, and participate in the consensus protocol. Consensus-layer rewards are predictable and constitute a low-variance incentive for block proposal.⁷ Execution-layer rewards, by contrast, are highly stochastic. The desire to maximize execution-layer rewards has driven the rise of a specialized and highly sophisticated ecosystem of Maximal Extractable Value⁸ (MEV) searchers, block builders, and relays [21].

Adversary. We assume a single adversary that controls a fraction α of the total network stake. All other validators are assumed to be under the control of honest proposers and will therefore propose in each available slot. The adversary makes a strategic choice to propose or miss in each slot that they are assigned. We will hereafter refer to the consecutive proposal slots that the adversary is assigned at the end of an epoch as the *tail* of the epoch⁹.

RANDAO Manipulation. Suppose the adversary controls a tail of length t in epoch E . Each subset of slots in which the adversary could propose will produce a unique RANDAO value at the end of the epoch.¹⁰ Because the adversary controls the validators assigned to the tail, and hence knows their private keys, the RANDAO value associated with each slot can be deterministically precomputed. Thus the adversary may sample 2^t distinct future values of the RANDAO corresponding to each subset of tail slots in which they could propose.

Modeling Assumptions. We make some assumptions while modeling the RANDAO mechanism:

- (a) Proposer sampling is uniformly random and the adversary gets selected to propose in a slot with α probability. Hence, each epoch assignment consists of 32 i.i.d. samples.
- (b) We ignore interleaving of epochs and assume that the RANDAO value at the end of an epoch determines the proposers for the next epoch.¹¹
- (c) Honest proposers always propose and the adversary always chooses to propose when they are in non-tail slots as it is in their best interest to do so.
- (d) We assume that the reward “resets” after an honest proposer, i.e., the reward distribution before and after an honest proposer is independent.

2.2 RANDAO Game

We now define the game induced by manipulating the RANDAO with an abstract reward model. Since general rewards may depend on the state in arbitrary ways, we begin by modeling the state space in its most general form as well. As an illustrative example, consider a scenario in which the adversary controls a tail of length 3. There are 2^3 different sequences of proposals that the adversary can make and one option is to do $\langle \text{propose}, \text{miss}, \text{propose} \rangle$ which suppose results in a future epoch assignment with tail 30 of the form $\langle \text{adversary}, \text{honest}, \text{adversary}, \dots \rangle$.

⁷ The rewards are not fixed, but depend on the total number of validators in the network. However, this number is relatively stable.

⁸ Maximal Extractable Value is defined as the reward available to a block proposer via their ability to selectively include, exclude, insert, and reorder transactions.

⁹ For example, denoting adversarial slots by “A” and honest slots as “H”, $(\dots H)$ has tail 0, $(\dots HA)$ has tail 1, $(\dots HAA)$ has tail 2, and so on.

¹⁰ Note that this is only the case because each slot in the tail is associated with a *unique* validator, each of which is under the control of the adversary. Thus the associated signatures are unique, meaning that the adversary has t distinct 256 bit strings which they may mix into the RANDAO.

¹¹ There is justification for this assumption in prior work [2], and it essentially does not affect the analysis for the adversarial stake range of interest ($\alpha < 1/2$) since each epoch contains at least one honest proposer with high probability.

In a reward model in which we only consider the number of blocks, this action will have a reward of 3 blocks plus getting to keep playing the game from a tail of 30. In the general case, however, there will be a known reward distribution that depends on the proposal sequence¹², and the adversary will first evaluate the expected reward of $\langle \text{propose}, \text{miss}, \text{propose} \rangle$ followed by the expected reward of $\langle \text{propose}, \text{honest proposal} \rangle$ and the subsequent value of playing the game from a tail of length 30.

Proposal strings. At the core of our formulation is the definition of two types of proposal strings.

- *tail proposal string (TAIL)*: $p_t \in \{0, 1\}^k$ represents the event in which out of k consecutive slots all assigned to the adversary to propose, the adversary proposed in exactly the indices with 1.
- *non-tail proposal string (NEXT)*: $p_n \in \{0, 1\}^k$ represents the event in which out of k consecutive slots, the adversary proposed in exactly the indices with 1, while honest validators proposed in the remaining indices with 0 (recall that we assume all honest parties always propose).

Intuitively, when deciding what option to choose, the adversary will consider the expected reward of these two events and the resulting new tail of the action.

Epoch assignments. Let $\mathcal{E}$ be the distribution of epoch assignments of the form $\{0, 1\}^{32}$ where each bit is identically and independently drawn with probability α . We will refer to the tail length of an epoch assignment $E \sim \mathcal{E}$ as $\text{tail}(E)$ and the non-tail proposal string as $\text{non-tail-proposals}(E)$.

Reward model. We define the rewards as an arbitrary function of the proposal strings representing the expected reward of the reward distribution. A reward model $\mathcal{R}$ is a pair $\langle \text{propose}, \text{next} \rangle$ where

- $\text{propose} : \{0, 1\}^* \rightarrow \mathbb{R}$ is a function that takes in a tail proposal string and outputs the expected reward of the adversary in taking this action.
- $\text{next} : \{0, 1\}^* \rightarrow \mathbb{R}$ is a function that takes in a non-tail proposal string and outputs the expected reward of the adversary in taking this action.

RANDAO game. Now, combining all the observations, we define the RANDAO manipulation game parameterized by adversarial stake α and reward model $\mathcal{R}$. The variables REWARD and ROUNDS count the accumulated reward by the adversary and the number of epochs that have passed. TAIL is used to keep track of the current tail controlled by the adversary.

(a) *Initialization.*

1. Let $\text{REWARD} := 0$ and $\text{ROUNDS} := 0$.
2. Let $\text{TAIL} := 0$.

(b) *Sampling.*

3. Let $E_P \sim \mathcal{E}$ for all $P \subseteq \{0, 1\}^{|\text{TAIL}|}$.
4. Let N_P be $\text{non-tail-proposals}(E_P)$ and similarly let T_P be $\text{tail}(E_P)$.

¹² One way to think about this is to consider the distribution of rewards that a particular adversary is able to obtain. For example, if we consider an adversary that is really good at extracting multi-block MEV, the reward distribution will depend on whether proposals are consecutive. Alternatively if there is not much correlation between block rewards, we can use a per-slot independent model. We consider these in more detail in Section 4.

- (c) *Strategic choice.*
5. The adversary chooses a subset P^* to propose in based on the reward model $\mathcal{R}$ and the set of sampled epochs.
- (d) *Updates.*
6. Let $\text{REWARD} := \text{REWARD} + \text{propose}_{\mathcal{R}}(P^*) + \text{next}_{\mathcal{R}}(N_{P^*})$.
 7. Let $\text{TAIL} = T_{P^*}$.
 8. Let $\text{ROUNDS} := \text{ROUNDS} + 1$.
 9. Repeat from step 3.

Objective. The optimization objective is $\liminf\{\text{REWARD}/\text{ROUNDS}\}$, the average reward in the infinite game.

3 Solving the RANDAO Game

In this section, we define the MDP formulation of the RANDAO game, using the state space reduction observation from [2]. The MDP formulation reduces the state space to just the tail lengths $\{0, 1, \dots, 32\}$, pushing the complexity of the game into the action space. The MDP solver in [2], however, does not immediately generalize to our case.

Diverging from [2], to give probable bounds on our sampling-based approach, we observe that the rare states can be truncated without significantly affecting the optimal reward. Concretely, we will define two truncations of the MDP where we ignore states above a certain threshold: (a) an *upper truncation* that rewards large tails state transitions maximally, and (b) a *lower truncation* that rewards large-tail state transitions 0. This will allow us to bound the optimal reward of the full MDP between the optimal rewards of the two truncated MDPs.

Comparison with [2]. Our MDP uses the same state space as [2], the tail length $t \in \{0, \dots, 32\}$. The main difference is in computing the transition probabilities. Computing each $P(t, t')$ requires summing, over the reward realizations the adversary could face from t , the probability that the adversary's optimal response leads to t' . Under uniform per-block rewards the sum has manageably few terms because the optimal response depends only on the *number of proposals*. Once the reward depends on the *pattern*, as in our examples, the sum has many more terms and quickly becomes intractable to evaluate. For example, the smooth multi-block model (Example 2) pays some function of k for each block of k consecutive proposals, so the sum has to enumerate over integer partitions of every proposal count $k \leq t$. Our methodology sidesteps this combinatorial issue by estimating the transitions via sampling, with an ε -approximation guarantee from a truncation argument and a concentration bound. As a practical benefit, new reward models can be plugged into the solver without re-deriving any closed forms.

3.1 Methodology

Observe that we can immediately write down the MDP for the RANDAO game with reward model $\mathcal{R}$. Let $\mathcal{S} = \{0, 1, \dots, 32\}$ be the state space, $\mathcal{A}$ be the set of valuation functions $v : \mathcal{S} \rightarrow \mathbb{R}$. Denote by $\pi : \mathcal{S} \rightarrow \mathcal{A}$ a policy. Let $P_\pi : \mathcal{S} \times \mathcal{S} \rightarrow [0, 1]$ be the transition probability such that $P_\pi(t, t')$ is the probability of transitioning from tail t to tail t' when the adversary picks the option that maximizes $\text{propose}(P) + \text{next}(N_P) + v(\text{tail}(E_P))$. Let $R_\pi : \mathcal{S} \rightarrow \mathbb{R}$ also be the expected reward of taking the action given by π in the game. Lastly,

let $\text{OPT}(\cdot)$ be the optimal reward of either the RANDAO MDP/game. Using a similar argument to [2], we know that we can reduce the state space to just the tail length by working with the RANDAO MDP.

► **Lemma 4.** $\text{OPT}(G) = \text{OPT}(M)$ where G is the RANDAO manipulation game and M is the RANDAO MDP.

As discussed at the start of this section, computing the transition probabilities for non-uniform rewards in closed form is not always tractable. We therefore propose sampling the transition probabilities to estimate them. While sampling transitions from large tails is expensive, we observe that we can truncate the MDP to small tails. Intuitively, as large tails are rare, ignoring them will not change the optimal reward by much.

Approximate RANDAO Solver.

1. Truncate the MDP to a state space $S^T = \{0, 1, \dots, T\}$ such that all transitions that go to a tail $t > T$ now transition to T . The reward of the transition is either X for upper truncation or 0 for lower truncation. We call this the T -truncation of the model^a.
2. Initialize values $v(\cdot) = 0$ for each state.
3. Empirically estimate the truncated transition probabilities and reward vector with N samples for each $P(t, \cdot)$ and $R(\cdot)$ to get matrix $\hat{P}$ and vector $\hat{R}$.
4. Compute average reward γ for $\langle \hat{P}, \hat{R} \rangle$ using the stationary distribution of $\hat{P}$ and taking the product with $\hat{R}$.
5. Perform a policy improvement step by solving for values using $V + \gamma = \hat{R} + \hat{P}V$ where V is the value vector, and set $v := V$. If values did not change more than the tolerance, output the current valuation. Otherwise continue from step 3.

^a We take $X = R_{\max} \cdot \mathbb{E}[L(T, \alpha)]$, an upper bound on the expected reward accrued during a large-tail sequence (see Lemma 5). As we show later, the upper and lower truncations give upper and lower bounds.

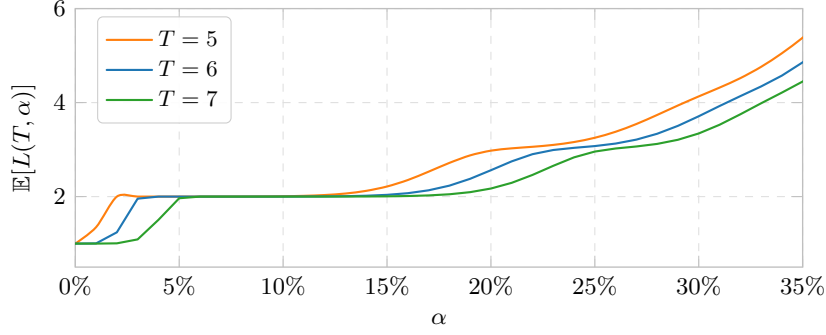
3.2 Reward Guarantees

We will now analyze the optimality loss of *truncation* and using the *approximate* model generated by samples. These bounds will ensure that we truncate conservatively and sample enough times to accurately approximate the optimal reward. Our analysis will depend on an upper bound on the maximum reward of a single transition $R_{\max}$. For example, if the reward is the number of blocks proposed, then $R_{\max} = 32$.

Truncation bound. Intuitively, we observe that any transition to a large tail $t > T$ must at some point transition back to a small tail $t \leq T$ for bounded α . Hence, we can bound the expected number of large tail transitions and assign the maximum reward $R_{\max}$ to each of these transitions to get an upper bound. To do this, we first need to bound the expected length of consecutive large tail transitions.

► **Lemma 5.** For any policy π , let $L^\pi(T, \alpha)$ denote the number of consecutive rounds the adversary spends in tails greater than T , starting from $t = 32$. Let $L(T, \alpha)$ be the corresponding quantity under the policy that always picks the longest tail. Then $L(T, \alpha)$ stochastically dominates $L^\pi(T, \alpha)$ for every π , and

$$\mathbb{E}[L(T, \alpha)] \leq X_{T, \alpha}$$



■ **Figure 1** Bound on $\mathbb{E}[L(T, \alpha)]$ from Lemma 5, plotted versus stake fraction α for truncation thresholds $T \in \{5, 6, 7\}$.

where $X_{T,\alpha}$ is the last element of the vector $(I - P_T)^{-1}\mathbf{1}$ where P_T is the transition matrix of the Markov chain with state space $\{T+1, \dots, 32\}$ and transition probabilities of the strategy that always picks the longest tail.

► **Theorem 6.** Let M^+ and M^- be the upper and lower T -truncations of the RANDAO MDP M . Then, $\text{OPT}(M^-) \leq \text{OPT}(M) \leq \text{OPT}(M^+)$

Sampling complexity. We now observe that by an application of the Chernoff-Hoeffding bound and a union bound over the $(T+1)^2$ entries of the transition matrix and the $(T+1)$ entries of the reward vector, we can derive the sufficient number of samples to estimate the probabilities and rewards up to an error of ε with probability $1 - \delta$.

► **Lemma 7.** Given $\varepsilon, \delta > 0$, let $\hat{P}$ and $\hat{R}$ be the empirical estimates of the transition probabilities and rewards using N independent samples per state from the truncated MDP M^T with a fixed policy where

$$N \geq \max \left(\frac{(T+1)^2}{2\varepsilon^2} \ln \left(\frac{4(T+1)^2}{\delta} \right), \frac{R_{\max}^2}{2\varepsilon^2} \ln \left(\frac{4(T+1)}{\delta} \right) \right)$$

Then, with probability at least $1 - \delta$, both of the following hold:

$$\sum_{t'} |P(t, t') - \hat{P}(t, t')| \leq \varepsilon, \quad \forall t \in \{0, \dots, T\}$$

$$|R(t) - \hat{R}(t)| \leq \varepsilon, \quad \forall t \in \{0, \dots, T\}$$

For $\varepsilon = 0.0085$ and $\delta = 0.01$, $N = 10^8$ samples per state is sufficient for $T = 7$ and the $R_{\max}$ of the reward models we consider. We will be using this setting in our experiments.

4 MEV Models and Results

In practice, MEV is a complex phenomenon that is affected by not just the number of blocks, but their successive combinations or the result of a stochastic process. We will therefore consider a few different MEV models that capture some of these phenomena. For all reported utilities, we plot the reward per slot (i.e. the average reward for a single slot) against α . We focus on stake fractions $\alpha \leq 0.35$ throughout: at larger α the expected time spent in large tails $\mathbb{E}[L(T, \alpha)]$ (Lemma 5, Figure 1) grows rapidly, and the truncation bound $R_{\max} \cdot \mathbb{E}[L(T, \alpha)]$ separating our upper and lower estimates degrades.

4.1 Warm-up: Reproducing Uniform Reward Results

To validate our methodology, we first reproduce the results of [2] in the case of uniform rewards. In this case, the reward model $\mathcal{R}$ is defined as the number of 1's in the proposal string for both `propose` and `next`. The results are shown in Table 1.

■ **Table 1** Comparison of the optimal rewards reported by [2] with our method's U (upper truncation) and L (lower truncation) bounds for uniform rewards (a single unit per proposed block). Each entry has an error of at most 0.0265% ($\varepsilon/32$).

α	5%	10%	15%	20%	25%	30%	35%
[2]	5.0483%	10.1881%	15.3996%	20.6777%	26.0247%	31.4516%	36.9734%
U	5.0483%	10.1879%	15.3998%	20.6783%	26.0323%	31.4916%	37.2290%
L	5.0483%	10.1878%	15.3997%	20.6776%	26.0250%	31.4487%	36.9581%

4.2 Per-slot MEV

We define the per-slot MEV reward model as follows (Example 1). In every slot, there is a base reward of 1 in two components:

- *Ephemeral reward*. This is the reward for proposing a block in that slot. If the block is not proposed, this reward is lost.
- *Rollover reward*. This is the reward that rolls over from the previous slot if it was not proposed. If the block is proposed, this reward is claimed. If the block is not proposed, this reward rolls over to the next slot.

Intuitively, there are several reasons why this model is reasonable. Certain consensus-layer block rewards are ephemeral and are lost if the current block is not proposed, while many execution-layer rewards (such as priority fees and arbitrage or liquidation opportunities) roll over to the next block if not claimed. Even some of the consensus rewards on Ethereum roll over: the block immediately following a missed slot contains approximately twice the proposer reward (see Figure 6) due to the additional attestations that can be included. We will use 1 reward per slot with 60% ephemeral and 40% rollover reward. Let $\text{next}(n_t)$ be defined simply as the number of 1s in p_{t+1} . Let $\text{propose}(p_t)$ be defined such that each proposed block gives a reward of $1.0 + r$, where r is the amount of rollover reward from previous slots. Each missed slot accumulates 0.4 rollover, and a proposal claims and resets it to 0. Using $R_{\max} = 32$ for this model, we run our approximate RANDAO solver and the results are shown in Table 2.

■ **Table 2** Optimal rewards for the per-slot MEV reward model. U (upper truncation) and L (lower truncation) are compared with the honest strategy H . Each entry has an error of at most 0.0265% ($\varepsilon/32$).

α	5%	10%	15%	20%	25%	30%	35%
U	5.0493%	10.1922%	15.4107%	20.7007%	26.0734%	31.5628%	37.3514%
L	5.0493%	10.1922%	15.4106%	20.6999%	26.0660%	31.5194%	37.0754%
H	4.9999%	10.0009%	15.0015%	20.0017%	25.0016%	30.0018%	35.0016%

4.3 Multi-block MEV

We now define the multi-block MEV reward models as follows. In every epoch, the rewards scale with the number of consecutive blocks proposed. We will consider two variants of this model:

- **Smooth Model.** The reward for proposing k consecutive blocks is $k\sqrt{k}$, a smooth superlinear shape that captures phenomena such as arbitrage across multiple blocks (Example 2).
- **Threshold Model.** If k consecutive blocks are proposed, the reward is k for $k < 5$ and $k + 10$ for $k \geq 5$. This captures phenomena such as oracle manipulation, where a proposer with a long enough run can manipulate an oracle to their advantage and gain a large reward (Example 3).

Both multi-block reward models above are stylized illustrations rather than calibrated estimates. The $k^{3/2}$ shape is loosely inspired by CFAMM arbitrage with fees [19]: fees create a no-trade band that the cross-venue price gap must cross before arbitrage becomes profitable, and in the regime relevant to short multi-block sequences the resulting option-like payoff can grow superlinearly in k . The threshold choice ($k = 5$, bonus 10) is similarly illustrative. Documented TWAP attacks have their sharp discontinuity at $k = 2$ [18]. Our purpose is to showcase a methodology that can quantify the impact of any such reward structure, should one emerge in practice for multi-block strategies that are not yet exploited at scale. The results for both models are shown below in Table 3.

■ **Table 3** Optimal rewards for multi-block MEV models. Results for the smooth reward model (top) and threshold reward model (bottom) are shown. U (upper truncation) and L (lower truncation) are compared with the honest strategy H .

Smooth Rewards							
α	5%	10%	15%	20%	25%	30%	35%
U	5.2616%	11.0913%	17.5533%	24.7437%	32.8185%	42.0835%	53.7479%
L	5.2616%	11.0913%	17.5529%	24.7392%	32.7769%	41.8354%	52.1490%
H	5.2072%	10.8585%	17.0013%	23.6931%	31.0040%	39.0198%	47.8438%

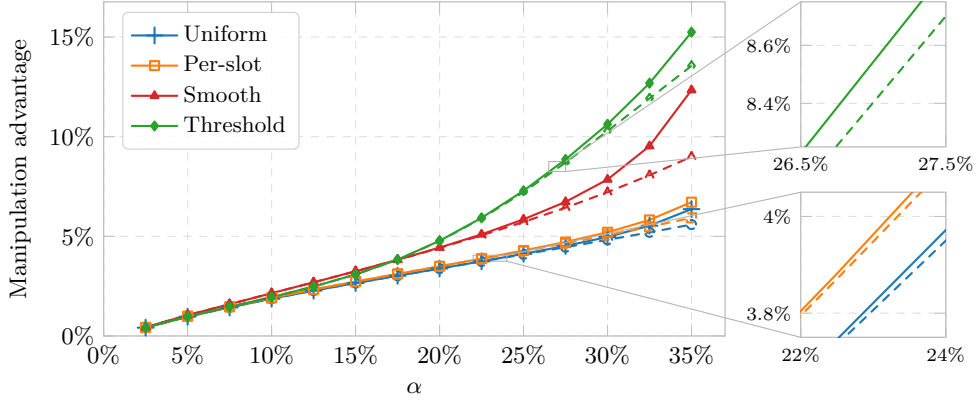
Threshold Rewards							
α	5%	10%	15%	20%	25%	30%	35%
U	5.0539%	10.2793%	15.8691%	22.1828%	29.7048%	38.9595%	50.7152%
L	5.0539%	10.2793%	15.8689%	22.1806%	29.6848%	38.8404%	49.9715%
H	5.0052%	10.0829%	15.3942%	21.1721%	27.6875%	35.2180%	44.0074%

5 Improvement over Honest Strategy

The severity of the incentive to manipulate RANDAO can be measured by the percentage increase in expected rewards per time over the honest strategy of proposing in every slot. We use this metric to compare the effects of different reward models on RANDAO manipulation incentives.

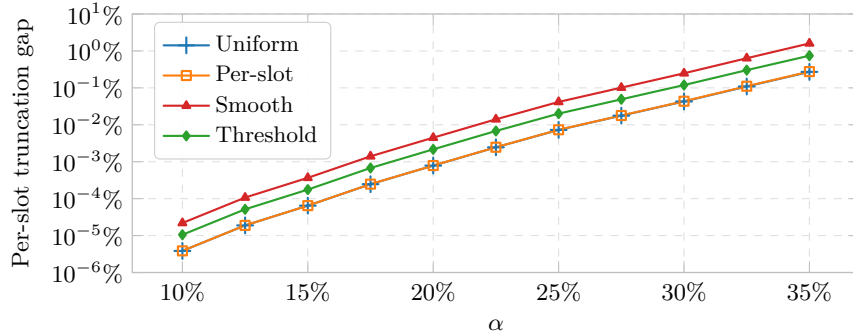
► **Definition 8 (Improvement).** Let $M^{\mathcal{R}}$ be a RANDAO MDP with reward model $\mathcal{R}$ and $h^{\mathcal{R}}$ be the average reward of the honest strategy in $M^{\mathcal{R}}$. Then, the improvement over the honest strategy is defined as: $\frac{\text{OPT}(M^{\mathcal{R}}) - h^{\mathcal{R}}}{h^{\mathcal{R}}}$

We find that this incentive is highly dependent on the reward model used as shown in Figure 2. This suggests that as the MEV extraction landscape evolves, the incentive to manipulate RANDAO may also change.



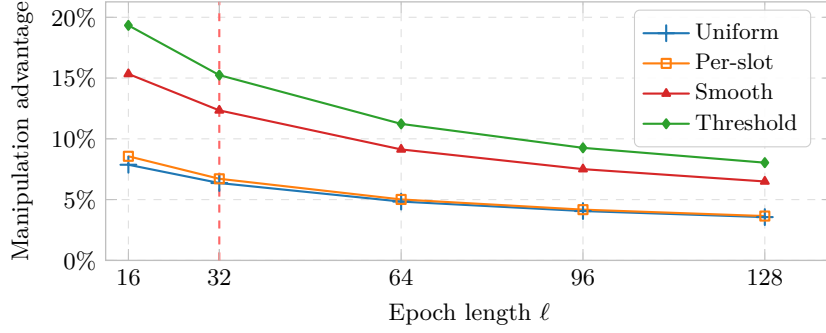
■ **Figure 2** Manipulation advantage over the honest strategy across reward models, measured as $(\text{OPT}(M^{\mathcal{R}}) - h^{\mathcal{R}})/h^{\mathcal{R}}$. Solid lines with filled markers show the upper-truncation bound. Dashed lines show the lower-truncation bound.

Error Analysis. We plot the absolute error between the upper and lower truncation in Figure 3. The error is small for small values of α , as expected, and the primary source of error is sampling. For larger values of α , the error increases as the optimal policy starts to depend more on the large tail states. However, even for $\alpha = 0.35$, the difference remains small. If higher precision is required, more samples and higher truncation T can be used.



■ **Figure 3** Per-slot gap between upper- and lower-truncation estimates of the optimal average reward at truncation tail length $T = 7$, across reward models and stake fractions α .

Epoch-Length Sensitivity. A natural protocol parameter to revisit is the epoch length ℓ itself, currently fixed at $\ell = 32$ slots on Ethereum mainnet. Let $M_{\ell}^{\mathcal{R}}$ denote the RANDAO MDP for reward model $\mathcal{R}$ under epoch length ℓ , and $h_{\ell}^{\mathcal{R}}$ the corresponding honest-strategy average reward. Figure 4 reports the relative manipulation gain $(\text{OPT}(M_{\ell}^{\mathcal{R}}) - h_{\ell}^{\mathcal{R}})/h_{\ell}^{\mathcal{R}}$ as we sweep ℓ , holding α fixed and computed separately for each of the four MEV reward models of Section 4. Across all four models, we observe that the manipulation incentive decays with increased epoch length. Lengthening the epoch is therefore a partial mitigation, but for any concrete ℓ the manipulation incentive depends on the reward model.



■ **Figure 4** Relative manipulation gain $(\text{OPT}(M_\ell^{\mathcal{R}}) - h_\ell^{\mathcal{R}})/h_\ell^{\mathcal{R}}$ versus epoch length ℓ at $\alpha = 0.35$, one curve per MEV reward model. The vertical red dashed line at $\ell = 32$ marks the current Ethereum value.

6 Tail Slashing

The preceding sections characterize how the incentive to manipulate RANDAO depends on the reward model. A natural next question is whether the reward structure itself can be modified to disincentivize manipulation. Introducing a missed-slot penalty in normal operation has been discussed within the Ethereum community [22]. In this section we apply our methodology in reverse: rather than computing the optimal adversarial strategy for a fixed reward model, we ask, for each MEV reward model, what is a minimal slashing curve that we can apply to the tail slots under which honest behavior becomes ε -optimal? We model tail slashing as a negative component of the reward function and reuse the solver.

6.1 Slashing as a Reward Model

We now pick an illustrative slashing scheme that penalizes tail misses. Let the epoch have length $\ell = 32$ and let $q \in \{0, 1\}^t$ denote the adversary's proposal pattern over a tail of length t , with $q_i = 1$ if the slot at position i from the end of the epoch was proposed ($i = 0$ is the last slot). Manipulation is driven by control of the final slots of an epoch, so we concentrate the penalty there. The penalty is driven by two factors: how close the missed slot is to the end of the epoch, and how many other misses occur in the same tail.

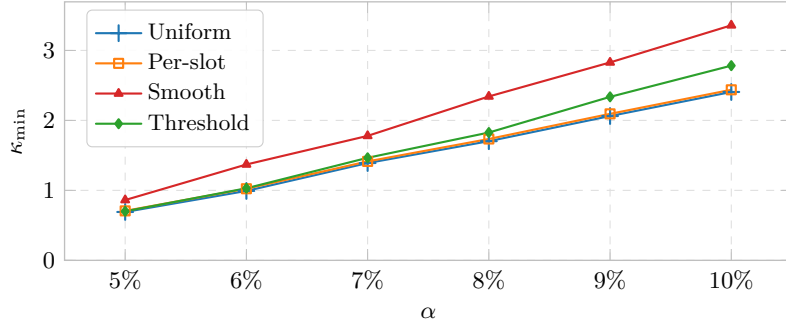
Slashing scheme. Consider the following slashing scheme parameterized by a scaling constant κ . For a fixed window $W = 8$, in every epoch, consider the last W slots. For each missed slot i in the tail (numbered such that $i = 0$ is the last slot), we will assign a slashing penalty of $\kappa \cdot \frac{W-i}{W} \cdot D$ where D is an additional multiplier equal to $1 + (\# \text{ of missed slots in the window})/W$. We run our experiments and discuss manipulability for this particular formula. For the game where every honest player proposes against a single α -adversary, we can write this as the following.

► **Definition 9 (Tail slashing).** Fix a window size $W \in \mathbb{N}$ and a slashing magnitude $\kappa \geq 0$. Index tail slots so that $i = 0$ is the last slot of the epoch. For a tail-proposal string q , define

$$S(q) = \sum_{i=0}^{\min(t, W)-1} \mathbf{1}[q_i = 0] \cdot \frac{W-i}{W}, \quad C(q) = |\{i < \min(t, W) : q_i = 0\}|.$$

The slashing penalty for q is

$$s_\kappa(q) = -\kappa \cdot \left(1 + \frac{C(q)}{W}\right) \cdot S(q)$$



■ **Figure 5** Slashing threshold $\kappa_{\min}$ versus stake fraction α , for each of the four MEV reward models of Section 4.

► **Example 10** (slashing with $W = 8$). We list the penalty for several tail-proposal patterns. Each binary string shows the proposal pattern over the window, read so the rightmost bit is the last slot of the epoch ($i = 0$).

$$\begin{aligned}
 11111110 &: s_{\kappa} = -\kappa \cdot \left(1 + \frac{1}{W}\right) \cdot 1 && \approx -1.125 \kappa \quad (\text{miss at the last slot}) \\
 01111111 &: s_{\kappa} = -\kappa \cdot \left(1 + \frac{1}{W}\right) \cdot \frac{1}{W} && \approx -0.141 \kappa \quad (\text{miss at } i = W-1) \\
 11111100 &: s_{\kappa} = -\kappa \cdot \left(1 + \frac{2}{W}\right) \cdot \frac{2W-1}{W} && \approx -2.344 \kappa \quad (\text{two clustered misses at end}) \\
 00000000 &: s_{\kappa} = -\kappa \cdot \left(1 + \frac{W}{W}\right) \cdot \frac{W+1}{2} && = -9 \kappa \quad (\text{all window slots missed})
 \end{aligned}$$

Slashed reward model. Given a reward model $\mathcal{R} = \langle \text{propose}, \text{next} \rangle$, the slashed model is

$$\mathcal{R}_{\kappa}^s = \langle \text{propose} + s_{\kappa}, \text{next} \rangle.$$

Slashing only modifies the **propose** component not **next**. We write $M^{\mathcal{R}_{\kappa}^s}$ for the RANDAO MDP (Section 3) instantiated with $\mathcal{R}_{\kappa}^s$.

► **Definition 11** (Slashing threshold). *For reward model $\mathcal{R}$, stake fraction α , and tolerance $\varepsilon > 0$, the slashing threshold $\kappa_{\min}$ is the minimum slashing magnitude $\kappa \geq 0$ for which honest behavior is ε -optimal in the slashed MDP $M^{\mathcal{R}_{\kappa}^s}$. That is, the optimal reward $\text{OPT}(M^{\mathcal{R}_{\kappa}^s})$ is at most ε above the honest average reward $h^{\mathcal{R}}$.*

The following monotonicity ensures that $\kappa_{\min}$ is well-defined.

► **Lemma 12** (Monotonicity in κ). *$\text{OPT}(M^{\mathcal{R}_{\kappa}^s})$ is non-increasing in κ .*

To certify that honest behavior is ε -optimal under a given κ , we need an upper bound on $\text{OPT}(M^{\mathcal{R}_{\kappa}^s})$. We therefore use only the upper truncation M^+ when computing $\kappa_{\min}$, and run at $N = 10^7$ samples per state.

6.2 Slashing Magnitudes

Figure 5 reports $\kappa_{\min}$ for the four MEV reward models of Section 4 across stake fractions α from 0.05 to 0.10. We observe that the ordering of $\kappa_{\min}$ at fixed α across MEV reward models matches the ordering of the manipulation gain $\text{OPT}(M^{\mathcal{R}}) - h^{\mathcal{R}}$ from Section 5: reward models that incentivize manipulation more strongly require larger penalties to neutralize as expected. Similarly, $\kappa_{\min}$ is monotonically increasing in α , since larger adversaries gain more from manipulation and require correspondingly larger slashing magnitudes.

Solo-staker friendliness. One concern for introducing a missed slot penalty is the effect on solo stakers. Solo stakers in practice miss only a small fraction of their assigned slots (roughly 1–2%), as i.i.d. accidents rather than coordinated drops at the tail. Under such a miss pattern, our slashing curve is far weaker on such validators than a fixed per-slot penalty that is independent of tail position and clustering. Misses early in the epoch incur little or no penalty under the decaying weight S , and the multi-miss multiplier stays close to 1 for isolated misses. A uniform per-slot penalty, by contrast, would charge the same κ for every accidental miss regardless of where in the epoch it falls or how it clusters with other misses.

7 Conclusion

We have shown that the incentive to manipulate RANDAO is highly dependent on the MEV reward model. In particular, partial reward rollover and multi-block MEV both amplify the manipulation gain, while lengthening the epoch is a partial mitigation across all reward models we considered. We also showed that tail slashing, modeled as a reward function within our methodology, can make honest behavior ε -optimal. Our methodology can be used to analyze the incentives around RANDAO in a variety of settings. For instance, recent work [8] incorporates a RANDAO bias analysis in the design of a protocol that uses randomness. Our codebase and general approach can streamline such analyses for other applications with complex reward models.

Future work. Three directions remain open. First, Section 6 analyzes a single tail-slashing curve built from a decaying penalty and a multi-miss multiplier. A natural extension is to explore alternative slashing curves, which may achieve smaller costs for honest validators under the same deterrence guarantee. Second, our framework works at the epoch level and we assume that the adversary commits to an action for the entire epoch at once. However, in practice, the adversary may do even better by adapting their strategy based on the blocks they see in the current epoch online. Analyzing this adaptive strategy is an interesting direction for future work. Third, the empirical gap between our upper and lower bounds is much smaller than our sample-complexity bound predicts, suggesting room for a tighter analysis.

References

- 1 Kaya Alpturer, Kushal Babel, and Aditya Saraf. Timing games in responsive consensus protocols, 2025. doi:10.48550/arXiv.2510.25144.
- 2 Kaya Alpturer and S. Matthew Weinberg. Optimal RANDAO Manipulation in Ethereum. In *AFT 2024*, 2024.
- 3 Musab A Alturki and Grigore Roşu. Statistical model checking of randao’s resilience to pre-computed reveal strategies. In *Formal Methods – FM 2019 International Workshops*, pages 337–349. Springer, 2019.
- 4 Kushal Babel, Philip Daian, Mahimna Kelkar, and Ari Juels. Clockwork finance: Automated analysis of economic security in smart contracts. In *2023 IEEE Symposium on Security and Privacy (SP)*, pages 2499–2516. IEEE, 2023. doi:10.1109/SP46215.2023.10179346.
- 5 Maryam Bahrani, Pranav Garimidi, and Tim Roughgarden. Transaction fee mechanism design in a post-mev world. In *6th Conference on Advances in Financial Technologies (AFT 2024)*, volume 316 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 29:1–29:24, 2024. doi:10.4230/LIPIcs.AFT.2024.29.
- 6 Maryam Bahrani, Michael Neuder, and S. Matthew Weinberg. Selfish mining under general stochastic rewards, 2025. doi:10.48550/arXiv.2502.20360.

- 7 Roi Bar-Zur, Ameer Abu-Hanna, Ittay Eyal, and Aviv Tamar. Werlman: To tackle whale (transactions), go deep (rl). In *2023 IEEE Symposium on Security and Privacy (SP)*, pages 93–110. IEEE, 2023. doi:10.1109/SP46215.2023.10179444.
- 8 Bhargav Nagaraja Bhatt, Fatemeh Shirazi, and Alistair Stewart. Trustless bridges via random sampling light clients. AFT 2025, 2025. URL: <https://eprint.iacr.org/2025/057>.
- 9 Linda Cai, Jingyi Liu, S. Matthew Weinberg, and Chenghan Zhou. Profitable Manipulations of Cryptographic Self-Selection Are Statistically Detectable. In Rainer Böhme and Lucianna Kiffer, editors, *6th Conference on Advances in Financial Technologies (AFT 2024)*, volume 316 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 30:1–30:23, Dagstuhl, Germany, 2024. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.AFT.2024.30.
- 10 Miles Carlsten, Harry Kalodner, S. Matthew Weinberg, and Arvind Narayanan. On the instability of bitcoin without the block reward. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, CCS ’16*, pages 154–167, New York, NY, USA, 2016. Association for Computing Machinery. doi:10.1145/2976749.2978408.
- 11 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, miner extractable value, and consensus instability. In *2020 IEEE symposium on security and privacy (SP)*, pages 910–927. IEEE, 2020. doi:10.1109/SP40000.2020.00040.
- 12 Data Always. mevboost-data: MEV-Boost relay bid archive. <https://github.com/dataalways/mevboost-data>.
- 13 Ben Edgington. *Upgrading Ethereum*. eth2book.info, Capella edition, 2023. URL: <https://eth2book.info/>.
- 14 Ittay Eyal and Emin Gün Sirer. Majority is not enough: Bitcoin mining is vulnerable. In *Financial Cryptography and Data Security*, pages 436–454. Springer, 2014. doi:10.1007/978-3-662-45472-5_28.
- 15 Matheus V. X. Ferreira, Ye Lin Sally Hahn, S. Matthew Weinberg, and Catherine Yu. Optimal strategic mining against cryptographic self-selection in proof-of-stake. In David M. Pennock, Ilya Segal, and Sven Seuken, editors, *EC ’22: The 23rd ACM Conference on Economics and Computation, Boulder, CO, USA, July 11 - 15, 2022*, pages 89–114. ACM, 2022. doi:10.1145/3490486.3538337.
- 16 Matheus V.X. Ferreira, Aadityan Ganesh, Jack Hourigan, Hannah Huh, S. Matthew Weinberg, and Catherine Yu. Computing optimal manipulations in cryptographic self-selection proof-of-stake protocols. In *EC ’24: The 25th ACM Conference on Economics and Computation*. ACM, 2024. doi:10.1145/3670865.3673602.
- 17 Flashbots and Robert Miller. Mev and the limits of scaling. <https://writings.flashbots.net/mev-and-the-limits-of-scaling>, 2025.
- 18 Torgin Mackinga, Tejaswi Nadahalli, and Roger Wattenhofer. Twap oracle attacks: Easier done than said? In *2022 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*, pages 1–8. IEEE, 2022. doi:10.1109/ICBC54727.2022.9805499.
- 19 Jason Milionis, Ciamac C Moallemi, and Tim Roughgarden. Automated market making and arbitrage profits in the presence of fees. In *International Conference on Financial Cryptography and Data Security*, pages 159–171. Springer, 2024. doi:10.1007/978-3-031-78676-1_9.
- 20 Ábel Nagy, János Tapolcai, István András Seres, and Bence Ladóczki. Forking the RANDAO: Manipulating ethereum’s distributed randomness beacon. Cryptology ePrint Archive, Paper 2025/037, 2025. URL: <https://eprint.iacr.org/2025/037>.
- 21 Burak Öz, Danning Sui, Thomas Thiery, and Florian Matthes. Who Wins Ethereum Block Building Auctions and Why? In Rainer Böhme and Lucianna Kiffer, editors, *6th Conference on Advances in Financial Technologies (AFT 2024)*, volume 316 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 22:1–22:25, Dagstuhl, Germany, 2024. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.AFT.2024.22.

- 22 Max Resnick. Missed slot penalties: temperature check, 2024. ethresear.ch forum post. URL: <https://ethresear.ch/t/missed-slot-penalties-temperature-check/18713>.
- 23 Ayelet Sapirshtein, Yonatan Sompolinsky, and Aviv Zohar. Optimal selfish mining strategies in bitcoin. In *Financial Cryptography and Data Security - 20th International Conference, FC 2016, Christ Church, Barbados, February 22-26, 2016, Revised Selected Papers*, pages 515–532, 2016. doi:10.1007/978-3-662-54970-4_30.
- 24 Caspar Schwarz-Schilling. Retroactive proposer rewards. Ethereum notes. URL: <https://notes.ethereum.org/@casparschwa/S1vcyXZL9>.
- 25 Caspar Schwarz-Schilling, Fahad Saleh, Thomas Thiery, Jennifer Pan, Nihar Shah, and Barnabé Monnot. Time is money: Strategic timing games in proof-of-stake protocols. In *5th Conference on Advances in Financial Technologies (AFT 2023)*, volume 282 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 30:1–30:17, 2023. doi:10.4230/LIPIcs.AFT.2023.30.
- 26 Pascal Stichler. Does multi-block MEV exist? Analysis of 2 years of MEV data, August 2024. URL: <https://ethresear.ch/t/does-multi-block-mev-exist-analysis-of-2-years-of-mev-data/20345>.
- 27 János Tapolcai, Bence Ladóczki, and Ábel Nagy. Slot a la carte: Centralization issues in ethereum’s proof-of-stake protocol. Cryptology ePrint Archive, Paper 2025/219, 2025. URL: <https://eprint.iacr.org/2025/219>.
- 28 Toni Wahrstätter. Selfish mixing and randao manipulation, July 2023. URL: <https://ethresear.ch/t/selfish-mixing-and-randao-manipulation/16081>.
- 29 Tao Yan, Shengnan Li, Benjamin Kraner, Luyao Zhang, and Claudio J. Tessone. Replication Data for: “Analyzing Reward Dynamics and Decentralization in Ethereum 2.0: A Data Engineering Workflow and Datasets”. Harvard Dataverse, V1, 2025. doi:10.7910/DVN/HG36L0.
- 30 Roi Bar Zur, Ittay Eyal, and Aviv Tamar. Efficient mdp analysis for selfish-mining in blockchains. In *Proceedings of the 2nd ACM Conference on Advances in Financial Technologies, AFT ’20*, pages 113–131, New York, NY, USA, 2020. Association for Computing Machinery. doi:10.1145/3419614.3423264.

A Proofs for Section 3 (Solving the RANDAO Game)

► **Lemma 4.** $\text{OPT}(G) = \text{OPT}(M)$ where G is the RANDAO manipulation game and M is the RANDAO MDP.

Proof. We first observe that the game is ergodic, since each epoch has a non-zero probability of having samples with the same fixed epoch (or tail length). Hence, the optimal average reward is independent of the starting state.

Now consider the RANDAO game G . Note that every epoch has a unique tail length. For any strategy playing the game, there is a corresponding strategy for the MDP that induces the same sequence of tails. As the game is ergodic, each state pair will be visited infinitely often, and in expectation these strategies will have the same average reward. Similarly, any strategy playing the MDP has a corresponding strategy in the game as the actions are the same. Hence, the optimal average reward must be the same. ◀

► **Lemma 5.** For any policy π , let $L^\pi(T, \alpha)$ denote the number of consecutive rounds the adversary spends in tails greater than T , starting from $t = 32$. Let $L(T, \alpha)$ be the corresponding quantity under the policy that always picks the longest tail. Then $L(T, \alpha)$ stochastically dominates $L^\pi(T, \alpha)$ for every π , and

$$\mathbb{E}[L(T, \alpha)] \leq X_{T, \alpha}$$

where $X_{T,\alpha}$ is the last element of the vector $(I - P_T)^{-1}\mathbf{1}$ where P_T is the transition matrix of the Markov chain with state space $\{T+1, \dots, 32\}$ and transition probabilities of the strategy that always picks the longest tail.

Proof. We start by observing that the strategy that always picks the longest tail stochastically dominates any other strategy in terms of the sequence of rounds spent in tails greater than T . Consider the next epoch samples across strategies: the tail-maximizing choice realizes a next-state tail length that is at least as large as any other strategy's. Moreover, choosing the max tail dominates returning to smaller states. Hence, the expected length of consecutive rounds in large tails of the tail-maximizing strategy is an upper bound on the expected length of consecutive rounds in large tails of any strategy. Lastly, we observe that the length of this path is maximized by starting from the largest tail 32. Hence, as the expected length of rounds in large tails starting from each large state is given by the vector $(I - P_T)^{-1}\mathbf{1}$, the expected length starting from 32 is the last element of this vector. $\blacktriangleleft$

► **Theorem 6.** Let M^+ and M^- be the upper and lower T -truncations of the RANDAO MDP M . Then, $\text{OPT}(M^-) \leq \text{OPT}(M) \leq \text{OPT}(M^+)$

Proof. M^- differs from M only by replacing the reward on each transition to a large tail with 0, so any policy's average reward in M^- is at most its average reward in M . Hence, $\text{OPT}(M^-) \leq \text{OPT}(M)$. Now consider the optimal policy π^* for M . Observe that whenever π^* transitions to a large tail $t > T$, it must eventually transition back to a small tail $t \leq T$. As this will last less than $L(T, \alpha)$ rounds in expectation, we can upper bound the expected reward of this path by $R_{\max} \cdot \mathbb{E}[L(T, \alpha)]$. Therefore, the corresponding policy in M^+ that takes the same actions as π^* in small tails and transitions to T with reward $R_{\max} \cdot \mathbb{E}[L(T, \alpha)]$ in large tails will have an expected reward at least as much as π^* . Moreover, since the M^+ policy collapses each large-tail sequence into a single transition while π^* takes $\mathbb{E}[L(T, \alpha)]$ steps in expectation, the M^+ policy accumulates at least as much reward over strictly fewer transitions, so its long-run average reward is at least that of π^* . Hence, $\text{OPT}(M) \leq \text{OPT}(M^+)$ as desired. $\blacktriangleleft$

► **Lemma 7.** Given $\varepsilon, \delta > 0$, let $\hat{P}$ and $\hat{R}$ be the empirical estimates of the transition probabilities and rewards using N independent samples per state from the truncated MDP M^T with a fixed policy where

$$N \geq \max \left(\frac{(T+1)^2}{2\varepsilon^2} \ln \left(\frac{4(T+1)^2}{\delta} \right), \frac{R_{\max}^2}{2\varepsilon^2} \ln \left(\frac{4(T+1)}{\delta} \right) \right)$$

Then, with probability at least $1 - \delta$, both of the following hold:

$$\sum_{t'} |P(t, t') - \hat{P}(t, t')| \leq \varepsilon, \quad \forall t \in \{0, \dots, T\}$$

$$|R(t) - \hat{R}(t)| \leq \varepsilon, \quad \forall t \in \{0, \dots, T\}$$

Proof. We proceed with applying a Chernoff-Hoeffding bound on each estimate and then taking a union bound over all the estimates. For each state t , we have N independent samples of the next state t' . Let X_i be the indicator random variable for the event that the i^{th} sample transitioned to t' . Then, $\hat{P}(t, t') = \frac{1}{N} \sum_{i=1}^N X_i$ and $\mathbb{E}[\hat{P}(t, t')] = P(t, t')$. By the Chernoff-Hoeffding bound, we have

$$\Pr \left[|\hat{P}(t, t') - P(t, t')| \geq \frac{\varepsilon}{T+1} \right] \leq 2 \exp \left(-\frac{2N\varepsilon^2}{(T+1)^2} \right)$$

Now, we know that $\sum_{t'} |P(t, t') - \hat{P}(t, t')| \leq \varepsilon$ if $|P(t, t') - \hat{P}(t, t')| \leq \varepsilon/(T+1)$ for all t' . Therefore, by a union bound over all t' , we have

$$\Pr \left[\bigcup_{t'} \left[|\hat{P}(t, t') - P(t, t')| \geq \frac{\varepsilon}{T+1} \right] \right] \leq 2(T+1) \exp \left(-\frac{2N\varepsilon^2}{(T+1)^2} \right)$$

Hence, by another union bound over all t , we have

$$\sum_{t'} |P(t, t') - \hat{P}(t, t')| \leq \varepsilon, \quad \forall t \in \{0, \dots, T\}$$

with probability at least $1 - 2(T+1)^2 \exp(-2N\varepsilon^2/(T+1)^2)$. As $N \geq \frac{(T+1)^2}{2\varepsilon^2} \ln \left(\frac{4(T+1)^2}{\delta} \right)$, this is $\geq 1 - \delta/2$.

Now consider the reward estimates. Let Y_i be the random variable representing the reward of the i^{th} sample from state t . Then, $\hat{R}(t) = \frac{1}{N} \sum_{i=1}^N Y_i$ and $\mathbb{E}[\hat{R}(t)] = R(t)$. By the Chernoff-Hoeffding bound, using the fact that $Y_i \in [0, R_{\max}]$, we have

$$\Pr \left[|\hat{R}(t) - R(t)| \geq \varepsilon \right] \leq 2 \exp \left(-\frac{2N\varepsilon^2}{R_{\max}^2} \right)$$

Since there are $T+1$ states, by a union bound over all t , we have

$$|R(t) - \hat{R}(t)| \leq \varepsilon, \quad \forall t \in \{0, \dots, T\}$$

with probability at least $1 - 2(T+1) \exp(-2N\varepsilon^2/R_{\max}^2)$. As $N \geq \frac{R_{\max}^2}{2\varepsilon^2} \ln \left(\frac{4(T+1)}{\delta} \right)$, this is $\geq 1 - \delta/2$.

Therefore, by a final union bound, we have that both properties hold with probability at least $1 - \delta$ as desired. $\blacktriangleleft$

B Proofs for Section 6 (Tail Slashing)

► **Lemma 12** (Monotonicity in κ). $\text{OPT}(M^{\mathcal{R}_{\kappa}^s})$ is non-increasing in κ .

Proof. Fix $\kappa < \kappa'$ and let π be optimal for $M^{\mathcal{R}_{\kappa'}^s}$. On every transition where π skips a tail slot, the slashing penalty under κ' is at least as large as under κ . On transitions with no skip, the reward is identical. Since both MDPs are ergodic, the long-run average reward of π is well-defined and starting-state independent in each. Hence the average reward of π in $M^{\mathcal{R}_{\kappa}^s}$ is at least its average reward in $M^{\mathcal{R}_{\kappa'}^s}$, giving $\text{OPT}(M^{\mathcal{R}_{\kappa}^s}) \geq \text{OPT}(M^{\mathcal{R}_{\kappa'}^s})$. $\blacktriangleleft$

C Reward plots

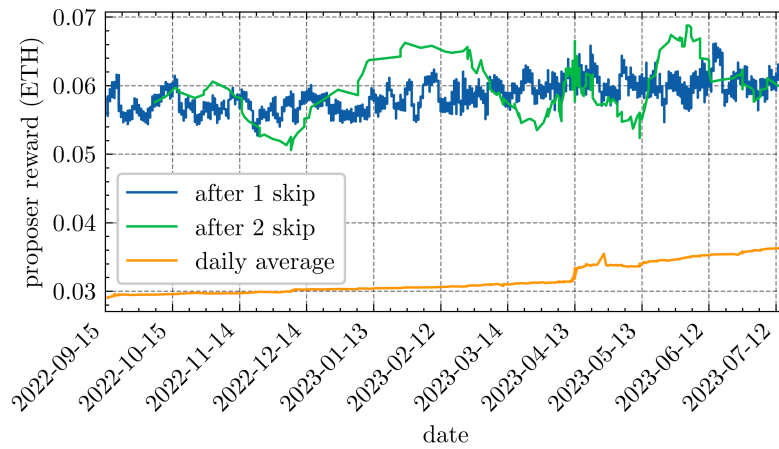


Figure 6 Consensus-layer rewards over time, excluding execution-layer rewards. Data from [29]. After a single missed slot, the next block earns roughly twice the consensus reward. Two or more consecutive missed slots show no significant further increase.

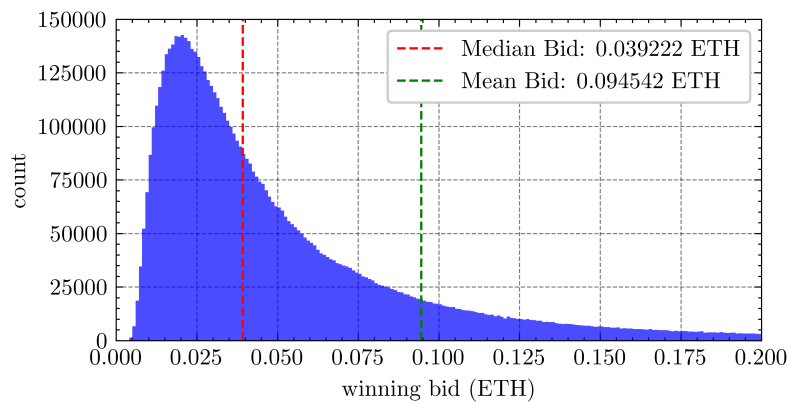


Figure 7 Histogram of bids from the MEV-Boost dataset [12].