Timestamp Manipulation: Timestamp-based Nakamoto-style Blockchains are Vulnerable

Junjie Hu
Department of Computer Science and Engineering
Shanghai Jiao Tong University
Shanghai 200240, China
Email: nakamoto@sjtu.edu.cn

Na Ruan
Department of Computer Science and Engineering
Shanghai Jiao Tong University
Shanghai 200240, China
Email: naruan@sjtu.edu.cn

***Abstract*—We introduce two advanced attack strategies, the Unrestricted Uncle Maker (UUM) Attack and the Staircase-Unrestricted Uncle Maker (SUUM) Attack, which fundamentally threaten the security of timestamp-based Nakamoto-style blockchains by inflicting permanent systemic harm. Unlike prior work that merely enhances adversarial rewards, these attacks exploit vulnerabilities in timestamp manipulation and fork selection rules to irreversibly destabilize blockchain fairness and incentive mechanisms. Specifically, the SUUM attack enables adversaries to persistently launch attacks at zero cost, eliminating constraints on block withholding and risk-free conditions, while systematically maximizing rewards through coordinated timestamp adjustments and strategic block release.**

Our analysis demonstrates that SUUM adversaries achieve disproportionate reward advantages over both UUM and the original Riskless Uncle Maker (RUM) Attack [CCS '23], with all three strategies surpassing honest mining. Crucially, SUUM's cost-free persistence allows adversaries to indefinitely drain rewards from honest participants by maintaining minimal difficulty risks through precise timestamp manipulation. This creates a self-reinforcing cycle: adversaries amplify their profits while suppressing honest returns, thereby permanently eroding the protocol's security assumptions. Through rigorous theoretical modeling and simulations, we validate how SUUM's combination of timestamp tampering, block withholding, and difficulty risk control enables unmitigated exploitation of consensus mechanisms. This work underscores the existential risks posed by timestamp-based Nakamoto-style protocols and advocates urgent countermeasures to ensure long-term stability.

1. Introduction

The Proof-of-Work (PoW) blockchains [1], [2], with Bitcoin [3] and Ethereum 1.x [4], [5] as prominent examples, have significantly propelled the thriving development of the cryptocurrency sector, which boasts a market valuation of 1.9 trillion dollars [6]. These advanced blockchain systems are built upon highly decentralized blockchain protocols and ingeniously employ the PoW mechanism to ensure their consistency and security [7], [8], [9], [10], [11], [12]. The blockchain network is jointly maintained and operated by a group of active participants known as miners, who possess and control a certain amount of computational resources, such as high-performance computers and specialized mining hardware. Participants invest substantial computational power to solve elaborately designed and extremely complex cryptographic puzzles, a process referred to as mining [13]. Participants who successfully solve these puzzles gain the right to record the next valid block on the blockchain.

The blockchain system provides generous incentives to participants who successfully generate valid blocks, as a reward for their diligent work and invested computational resources. These incentives typically consist of a certain quantity of newly issued cryptocurrencies and possibly transaction fees included in the block. Ideally, the reward allocation mechanism designed by the blockchain system is relatively equitable, accurately reflecting the proportion of computational resources invested by participants [14], [15], [16]. This implies that participants with more computational resources will have a higher probability of mining blocks and receiving corresponding coin-base rewards, thereby incentivizing them to continue contributing to the stable operation and security of the blockchain network. Numerous cryptocurrencies that emerged after Bitcoin and subsequent academic literature have proposed and implemented numerous schemes to alter the blockchain reward mechanism [17], [18], [19], [20]. The purpose of these changes is to ensure that participants have no incentive to manipulate the system for personal gain.

In September, 2022, Ethereum transitioned from its Proof-of-Work mechanism (known as Ethereum 1.x) to a new Proof-of-Stake mechanism (designated as Ethereum 2.x). It is noteworthy that Ethereum 1.x's Proof-of-Work version continues to be utilized by cryptocurrencies such as Ethereum PoW [21], Ethereum Fair [22] and Ethereum Classic [23]. These cryptocurrencies have evidently gained popularity among participants: as of April 1, 2025, their combined hash power accounted for 13.9% of the peak hash power observed in the Ethereum 1.x ecosystem [24]. Ethereum 1.x boasts a significantly reduced block time of approximately 13 seconds on average, compared to Bitcoin's average block time of 10 minutes, which results

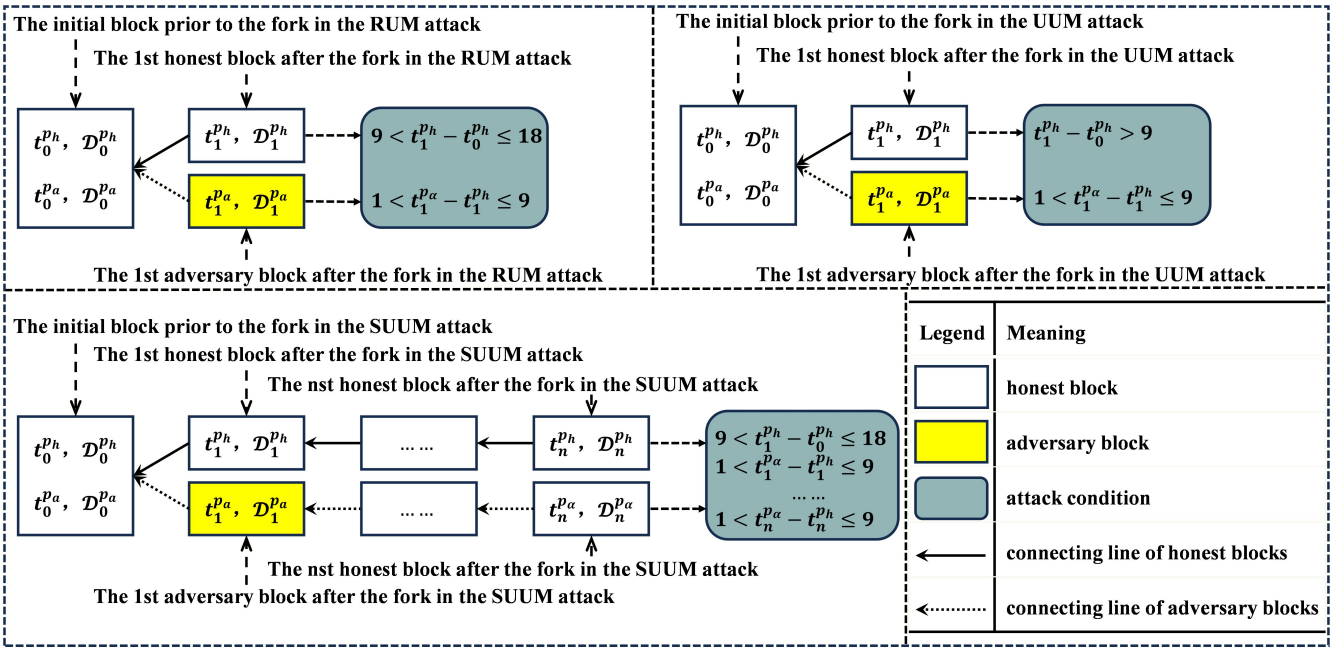


Figure 1. Attack Flowchart. This figure illustrates the attack flowchart for the proposed RUM, UUM and SUUM attacks in timestamp-based Nakamoto-style blockchains. The flowchart delineates the systematic process through which adversaries manipulate block timestamps and strategically withhold or release blocks to gain disproportionate rewards. It highlights the adversarial strategies' escalation from RUM (risk-constrained) to UUM (risk-tolerant) and SUUM (withholding-enabled), emphasizing their impact on blockchain protocol and incentive fairness.

in frequent state fork (also known as soft fork) [25]. To facilitate rapid convergence of blockchain states in the midst of forks, the Ethereum 1.x protocol mandates that honest participants select the branch with higher mining difficulty [4]. Consequently, a series of mining attacks aimed at maliciously manipulating block difficulty have been proposed and proven effective in reducing blockchain security [16], [26], [27], [28], [29], [30], [31]. Among them, the most renowned is the riskless Uncle Maker attack (hereinafter referred to as RUM), which grants adversaries a risk-free and highly lucrative advantage by meticulously manipulating block timestamps. By examining Ethereum 1.x blockchain data, the authors of RUM provided compelling evidence indicating that participants executed variants of the RUM attack for approximately two years [32]. One of the cofounders of F2Pool (then the second-largest mining pool for Ethereum 1.x) acknowledged their manipulation of block timestamps in this manner, marking the first instance of consensus protocol manipulation by participants in practice [33].

Our work introduces two advanced attack strategies, the Unrestricted Uncle Maker (UUM) Attack and the Staircase-Unrestricted Uncle Maker (SUUM) Attack, that fundamentally threaten the security of timestamp-based Nakamoto-style blockchains by inflicting permanent systemic harm. Unlike prior work that merely enhances adversarial rewards, these attacks exploit vulnerabilities in timestamp manipulation and fork selection rules to irreversibly destabilize blockchain fairness and incentive mechanisms. Specifically, the SUUM attack enables adversaries to persistently launch attacks at zero cost, eliminating constraints on block withholding and risk-free conditions, while systematically maximizing rewards through coordinated timestamp adjustments and strategic block release. By combining timestamp tampering with staggered block release, adversaries create a self-reinforcing cycle: they suppress honest participants' returns by manipulating difficulty growth rates while amplifying their own profits through persistent chain reorganizations. Through rigorous theoretical modeling and large-scale simulations, we validate that SUUM adversaries achieve disproportionate reward advantages (e.g., 33.30% vs 28.41% for UUM at $\alpha = 0.25$) over both honest mining and prior attacks, all while maintaining minimal difficulty risks through precise timestamp calibration. It is noteworthy that our proposed attacks are applicable to any blockchain system that adopts the Ethereum 1.x's fork selection rule, including Ethereum PoW [21], Ethereum Fair [22], Ethereum Classic [23], and others.

This asymmetric advantage arises from the synergistic exploitation of three core mechanisms: 1) Timestamp manipulation: Adversaries artificially inflate their block difficulty by retroactively adjusting timestamps to exploit Ethereum 1.x's fork selection rules. 2) Block withholding: SUUM adversaries strategically delay block releases to maximize chain reorg opportunities, creating persistent forks that disadvantage honest miners. 3) Difficulty risk control: By calibrating timestamps to the granularity of seconds, attackers suppress blockchain difficulty escalation, ensuring their

operations remain cost-free and sustainable indefinitely.

The cumulative effect of these strategies permanently erodes the protocol's security assumptions. Honest participants face diminishing returns as adversaries drain rewards proportionally, a zero-sum game where attackers' gains directly translate to systemic losses. Our simulations demonstrate that even adversaries with modest computational power (e.g., $\alpha = 0.3$) reduce honest participants' rewards by 11.85% compared to baseline honest mining. This imbalance escalates exponentially as adversarial power grows, ultimately leading to the collapse of the protocol's economic model, where rational participants are incentivized to join adversarial coalitions rather than mine honestly.

This work underscores the existential risks posed by timestamp-based Nakamoto-style protocols. The SUUM attack's cost-free persistence and irreversible damage necessitate urgent countermeasures. We advocate for protocol redesigns that decouple difficulty adjustments from adversarial timestamp control and introduce economic penalties for abnormal block intervals. Without such interventions, the self-reinforcing nature of SUUM-like attacks threatens the long-term viability of Ethereum 1.x derivatives and similar blockchain systems.

Our Contributions. We summarize the contributions of this paper as follows:

- **Synergistic Attack Vector Design.** We formalize a three-pillar framework combining 1) timestamp manipulation to inflate difficulty, 2) block withholding to maximize reorgs, and 3) difficulty risk control via temporal gap optimization. This synergy enables adversaries to systematically drain rewards while suppressing honest participation (Section 5.1, Theorem 6).
- **Permanent Protocol Harm.** We demonstrate that UUM and SUUM attacks irreversibly destabilize the foundational assumptions of timestamp-based Nakamoto-style protocols. By exploiting fork selection rules and difficulty adjustment mechanisms, adversaries create persistent reward distortions that persist even post-attack, eroding protocol fairness (Section 6.2).
- **Cost-free Attack Sustainability.** SUUM adversaries achieve cost-free persistence through second-level timestamp manipulation and strategic block withholding, eliminating traditional constraints like hash power thresholds (Theorem 7). Simulations confirm minimal difficulty escalation (0.21 maximal risk) despite sustained exploitation (Figure 6).
- **Systemic Economic Collapse.** We prove SUUM triggers a death spiral: adversarial rewards scale super-linearly (e.g., 43.7% at $\alpha = 0.4$), incentivizing rational miners to defect. This accelerates protocol abandonment, ultimately collapsing the economic model (Section 7.3).
- **Empirical Validation.** We implement a discrete-event timestamp-based Nakamoto-style blockchan simulator and conduct large-scale experiments (1M blocks, 10K trials) to quantify SUUM's dominance: adversaries with $\alpha = 0.3$ reduce honest rewards by 11.85%, while SUUM's forking rate (3.2%) exceeds UUM (2.1%) and RUM (1.4%) at equivalent power (Section 7.2, Figure 5-6).
- **Mitigation Framework.** We propose countermeasures including timestamp consensus mechanisms and difficulty decoupling protocols, establishing defense principles against Uncle Maker attacks (Section 8).

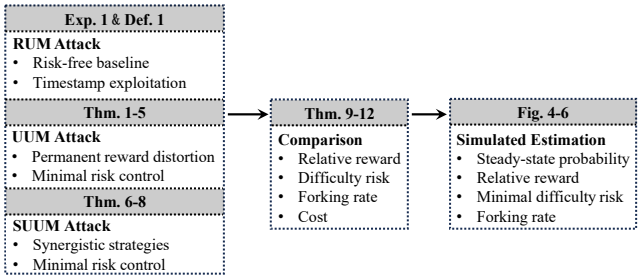


Figure 2. Roadmap. This figure outlines the research roadmap, starting with the foundational RUM attack as a baseline, which exploits timestamp manipulation for risk-free reward extraction. It progresses to the UUM attack, relaxing risk constraints to enable more frequent attacks, and culminates in the SUUM attack, which introduces strategic block withholding to form a three phase synergistic strategy-timestamp manipulation, block withholding, and difficulty risk control. The roadmap highlights key theorems (Thm. 1-12) that formalize attack conditions, state transitions, and reward dynamics, paired with simulated estimation of steady-state probabilities, relative rewards, and forking rates to validate SUUM's dominance.

Paper Structure. The structure of this paper is as follows: we first present the threat model of the system, including the composition of participants, adversary's target, honest participant's strategy space and adversary's strategy space in Section 2. Secondly, we review the design of the original RUM attack, provide formal definitions for block difficulty and block rewards, and illustrate fork selection and blockchain variable calculations through two examples in Section 3. Thirdly, we propose the advanced variant of the RUM attack, named UUM, and comprehensively analyze UUM from four aspects: method, state space, state transition, and reward analysis in Section 4. Fourthly, based on UUM, we introduce an even more advanced variant, SUUM, and similarly conduct a comprehensive analysis of SUUM from the aspects of method, state space, state transition, and reward analysis in Section 5. Fifthly, we compare RUM, UUM, SUUM, and honest mining from the perspective of reward, difficulty risk, forking rate, and attack cost in Section 6. Sixth, we validate theoretical claims through large-scale simulations, quantifying steady-state probabilities, reward disparities, and risk impacts in Section 7. Finally, we explore countermeasures to incentivize against Uncle Maker-based attacks in Section 8. Semi-formally, the roadmap in this paper is shown in Figure 2.

2. Threat Model

Participant Composition. We assume a decentralized protocol, denoted as Γ, which is collectively executed by a set of participants, $\mathcal{P}$, across consecutive time slots. The set of participants, $\mathcal{P}$, can be subdivided into two subsets: the

set of honest participants, $\mathcal{P}_{\mathcal{H}} = \{p_h^1, p_h^2, \ldots, p_h^n\}$, who strictly adhere to the protocol Γ, and the set of adversaries, $\mathcal{P}_{\mathcal{A}}$, who potentially violate the protocol. Consequently, the entire set $\mathcal{P}$ can be represented as the union of $\mathcal{P}_{\mathcal{H}}$ and $\mathcal{P}_{\mathcal{A}}$, i.e., $\mathcal{P} = \mathcal{P}_{\mathcal{H}} \cup \mathcal{P}_{\mathcal{A}}$.

It is noteworthy that the scope of honest mining participants, $\mathcal{P}_{\mathcal{H}}$, is not limited to a single entity. Its composition can be extended to mining pool organizations and even consortia formed by multiple mining pools. For the purpose of facilitating theoretical analysis and discussion, this paper uniformly abstracts such entities into the concept of a "single participant". Correspondingly, the adversary set, $\mathcal{P}_{\mathcal{A}}$, may also exist in the form of mining pools or mining pool alliances, exhibiting considerable complexity and diversity. This paper follows the common assumptions in the blockchain literature [4], [9], [34], namely, that during the attack period, the hash power of each participant remains constant, no new participants join the network, all mining hardware is preconfigured, and relevant cost (including but not limited to electricity expenses and mining hardware acquisition cost) are prepaid. In practice, historical data indicate that the active hashing rate in the network remains relatively stable over short periods, with minor fluctuations [35].

In protocol Γ, each participant $p \in \mathcal{P}$ is associated with a numerical value μ_p that lies between 0 and 1, such that $\sum_{p\in\mathcal{P}} \mu_p = 1$. Here, μ_p represents the participation power ratio of participant p in protocol Γ, which can be embodied specifically as hash power, stake power, or other relevant metrics.

Regarding the settings of timestamps and block structures, we assume that the timestamp of the genesis block $\mathcal{B}_0$ is $t_0 = 0$. Starting from the genesis block, the timestamp of block $\mathcal{B}_i$, which is generations of i away from the genesis block on the main chain, is denoted as t_i, with a corresponding difficulty of $\mathcal{D}_i$. Its immediate predecessor (parent block) $\mathcal{B}_{i-1}$ has a timestamp of $t_i^p = t_{i-1}$. Additionally, a variable $pu_i \in \{0, 1\}$ is introduced to indicate whether the parent block of block $\mathcal{B}_i$ references any uncle blocks. If it does, then $pu_i = 1$; otherwise, $pu_i = 0$.

Adversary's Target. Ethereum 1.x, as a cryptocurrency system, embraces the same notion of fairness as its counterparts, namely, participants with an expected hash rate proportion of μ_p should be able to mine a corresponding proportion of μ_p blocks and thus obtain μ_p of the mining rewards, as reported in [9], [34]. In this study, the core target of the adversary $\mathcal{P}_{\mathcal{A}}$ is to surpass their fair share based on hash rate, specifically by mining more blocks than their proportional share $\mu_{\mathcal{P}_{\mathcal{A}}}$, thereby obtaining rewards exceeding their deserved portion. The achievement of this target will be verified through subsequent theoretical analysis.

Honest Participant's Strategy Space. The definition of honest mining protocol in this paper refers to the relevant academic literature in the field of blockchain [34], [36] and specifically follows the rules established by the Ethereum 1.x White Paper [4]. Accordingly, honest participants always strive to mine the blocks with the highest total difficulty and strictly follow the protocol requirements, neither withholding blocks nor engaging in any form of manipulation of block timestamps.

Adversary's Strategy Space. In contrast, adversaries possess greater freedom in selecting their strategies. Within the framework of this study, adversaries are permitted to deviate from the honest mining protocol to a certain extent, but the blocks they produce must strictly comply with the validity requirements of the Ethereum 1.x protocol rules [4]. Specifically, adversaries are free to adjust the timestamps of the blocks they mine, but must ensure that these timestamps fall within a valid range (as described in Definition 1). Furthermore, adversaries have complete autonomy in choosing which blocks to mine.

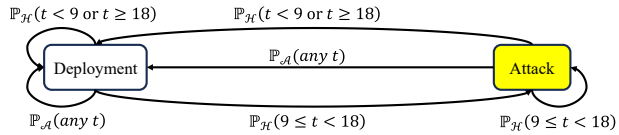
To clearly distinguish the attack models proposed in this study from the attack strategies presented in previous literature [34], [36], [37], [38], [39], we introduce two specific types of adversaries: UUM and SUUM. The UUM adversary abandons the risk-free constraint of the RUM attack, increases the steady-state probability of being in the attack state, and possesses the ability to selectively set block timestamps, but is not allowed to withhold blocks. Once a valid block is found by the UUM adversary, it must be immediately released. The SUUM adversary further relaxes the constraints by discarding the rule against withholding blocks in UUM, thereby further increasing its steady-state probability of being in the attack state. Meanwhile, the SUUM adversary retains the right to strategically set block timestamps.

3. The Design of RUM Attack

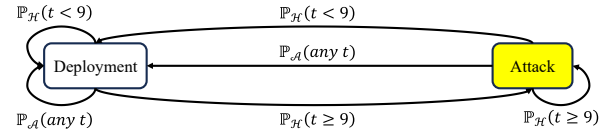
In this section, we introduce the RUM attack, which targets vulnerabilities in the incentive mechanism of the Nakamoto-style protocol. The target of this attack is for a RUM adversary who deviates from the protocol to obtain higher rewards than an honest participant who strictly adheres to the protocol specifications. We begin by reviewing the original RUM attack and, subsequently, build upon it to propose a low-risk, high-reward UUM attack. This attack removes the risk-free constraint, further increasing the probability of an adversary initiating the UUM attack and thus enhancing the utility of the UUM adversary. Finally, we further propose the SUUM attack based on this foundation. This attack lifts the restriction on adversaries withholding blocks, significantly expanding the adversary's strategic space and substantially boosting their utility. The only cost is a slight increase in block difficulty.

3.1. Preliminaries

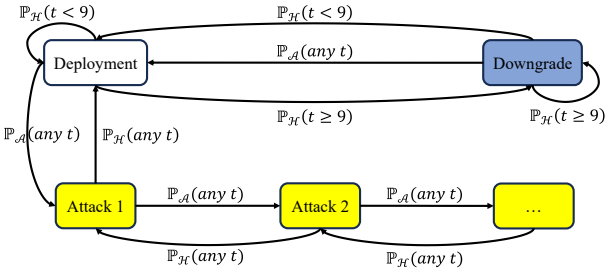
Each participant $\mathcal{P}$ in the Proof-of-Work blockchain engages in an iterative process aimed at deriving solutions to the cryptographic puzzle required for constructing a valid block. This process can be rigorously abstracted as a paradigm of Bernoulli trials: participants propose a solution randomly, and if this solution meets the established criteria, the trial outcome is recorded as true; otherwise, it is recorded as false. This series of mutually independent Bernoulli trials



(a) RUM state transition. In the RUM attack, the adversary employs a risk-free strategy and is not allowed to withhold blocks. The risk-free condition is achieved through the transition from the deployment state to the attack state when $t_1^{p_h} - t_0^{p_h} \in [9, 18)$.



(b) UUM state transition. Building upon the RUM framework, the UUM attack amplifies the steady-state probability of transitioning to the attack state, trading minimal risk for higher rewards. Specifically, the condition for transitioning from the deployment state to the attack state is extended to $t_1^{p_h} - t_0^{p_h} \in [9, +\infty)$. During the attack state, once the UUM adversary successfully discovers a new block, the attack is deemed successful by strategically setting the timestamp $t_1^{p_a} - t_0^{p_a} \in [1, 900)$.



(c) SUUM state transition: Building upon the UUM framework, the SUUM attack further enhances the adversary's strategy space by allowing the withholding of blocks. The Markov process of SUUM can be decomposed into two parts. On one hand, from the local perspective of the deployment and downgrade states, SUUM reduces to UUM, where the strategies of the two adversaries are identical. On the other hand, from the local perspective of the deployment and attack states, SUUM resembles traditional selfish mining, with the sole distinction being that the SUUM adversary meticulously manipulates the timestamp $t_i^{p_h} - t_i^{p_a} \in [1, 9)$ to ensure the success of the SUUM attack.

Figure 3. State Transition Process. This figure illustrates the state transition process under different mining strategies, delineating the dynamic transformation relationships among different states in the blockchain system. The diagram clearly presents the path from the initial state to the attack state through nodes and arrows, including the critical transition conditions between the deployment state and the attack state. It annotates the probabilities of state transitions and the behaviors of participants, revealing how attack strategies influence the blockchain's difficulty and reward distribution by manipulating timestamps and block release timing.

collectively constitutes a Bernoulli process. Upon observing this series of trials, the frequency of attempts required to achieve a successful outcome follows a geometric distribution. The duration spent on successfully discovering a valid block is allocated according to an exponential distribution.

Notably, both the geometric and exponential distributions exhibit the property of memorylessness. This implies that the success probability of each trial remains constant and is regulated by the difficulty parameter of the aforementioned protocol. Therefore, the probability of a participant finding a valid solution does not fluctuate due to their previous failures. Once any participant successfully obtains a valid block, by integrating it into their local blockchain and restarting the mining process, their opportunity to mine subsequent blocks remains unchanged. This property is also referred to as progress-free.

The RUM attack exploits the fact of the protocol Γ that honest participants $\mathcal{P}_A$ default to selecting the block with higher difficulty in the presence of fork competition in Ethereum 1.x. For ease of understanding, we illustrate an example of fork selection through Example 1.

We denote the timestamp and difficulty of the i-th block $\mathcal{B}_i^{p_h}$ on the honest branch during a fork competition as $t_i^{p_h}$ and $\mathcal{D}_i^{p_h}$, respectively. Similarly, the timestamp and difficulty of the i-th block $\mathcal{B}_i^{p_a}$ on the adversary's branch during a fork competition are denoted as $t_i^{p_a}$ and $\mathcal{D}_i^{p_a}$.

Example 1 (Fork Selection). *The blockchain comprises three blocks, denoted as $\mathcal{B}_0$, $\mathcal{B}_1^{p_a}$, and $\mathcal{B}_1^{p_h}$. The timestamp of block $\mathcal{B}_0$ is represented by $t_0^{p_h}$ or $t_0^{p_a}$, and its difficulty is denoted by $\mathcal{D}_0^{p_h}$ or $\mathcal{D}_0^{p_a}$. For block $\mathcal{B}_1^{p_a}$, the timestamp is $t_1^{p_a}$ and its difficulty is $\mathcal{D}_1^{p_a}$. Similarly, for block $\mathcal{B}_1^{p_h}$, the timestamp is $t_1^{p_h}$ and its difficulty is $\mathcal{D}_1^{p_h}$. According to the Ethereum 1.x protocol, participants are instructed to select the block with the maximum difficulty. Specifically:*
(1) Case 1: If $\mathcal{D}_1^{p_h} = max\{\mathcal{D}_1^{p_h}, \mathcal{D}_1^{p_a}\}$, select block $\mathcal{B}_1^{p_h}$.
(2) Case 2: If $\mathcal{D}_1^{p_a} = max\{\mathcal{D}_1^{p_h}, \mathcal{D}_1^{p_a}\}$, select block $\mathcal{B}_1^{p_a}$.
(3) Case 3: If $\mathcal{D}_1^{p_h} = \mathcal{D}_1^{p_a}$, select either block arbitrarily.

In the previously introduced attack scenarios, such as an adversary waiting for the emergence of a new block with a specific timestamp difference in the RUM attack, the objective is to leverage the block difficulty calculation rules under certain conditions, thereby giving the blocks they mine a competitive edge in the difficulty competition. Therefore, a thorough understanding of the principles and rules underlying block difficulty calculation is of significant importance for analyzing and defending against various attack behaviors targeting the Ethereum 1.x blockchain. Now, we provide a formal definition of block difficulty.

Definition 1 (Block Difficulty). *The difficulty $\mathcal{D}_i$ of block $\mathcal{B}_i$ satisfies the following equation:*

$$\mathcal{D}_i \overset{def}{=} max\left\{2^{17}, \mathcal{D}_i^p + f \cdot \left\lfloor \frac{\mathcal{D}_i^p}{2048} \right\rfloor\right\},$$

where $f = max\left\{1 + pu_i - \left\lfloor \frac{t_i - t_{i-1}}{9} \right\rfloor, -99\right\}$.

Please note that in Ethereum 1.x, the difficulty of blocks after height 15 exceeds 2^{17}. Consequently, in subsequent analyses concerning block difficulty, the difficulty $\mathcal{D}_i$ of block $\mathcal{B}_i$ is determined by the following equation:

$$\mathcal{D}_i \overset{def}{=} \mathcal{D}_i^p + max\left\{1 + pu_i - \left\lfloor \frac{t_i - t_{i-1}}{9} \right\rfloor, -99\right\} \cdot \left\lfloor \frac{\mathcal{D}_i^p}{2048} \right\rfloor.$$

3.2. Method

Based on the previously established rules for calculating block difficulty, it is evident that the difficulty of a block depends on the difficulty of its parent block and the difference in timestamps between the two blocks. An adversary can

exploit this rule by meticulously and manually manipulating the block timestamps to make their own block's difficulty higher than that of blocks on other branches, thereby increasing the likelihood of their block being selected preferentially. The RUM attack leverages this vulnerability. We show the state transition process of RUM in Figure 3-(a).

Specifically, the RUM attack manipulates block timestamps in two phases to create a difficulty advantage and achieve risk-free block prioritization. In the deployment phase, the adversary monitors blocks generated by honest participants and waits for the timestamp difference between a new honest block $\mathcal{B}_1^{p_h}$ and the previous mainchain block $\mathcal{B}_0^{p_h}$ to fall within the interval $t_1^{p_h} - t_0^{p_h} \in (9, 18]$ seconds (i.e., $\left\lfloor \frac{t_1^{p_h} - t_1^{p_h}}{9} \right\rfloor = 1$). This condition ensures the honest block is valid and that the adversary's mining difficulty on the parent block $\mathcal{B}_0^{p_h}$ matches the honest mining difficulty on $\mathcal{B}_1^{p_h}$, eliminating additional risk. Once this timestamp difference is met, the attack progresses to the execution phase.

During the execution phase, the adversary mines on the parent block $\mathcal{B}_0^{p_h}$ (identical to $\mathcal{B}_0^{p_h}$ and aims to generate a valid block $\mathcal{B}_1^{p_h}$ before honest participants. The adversary ensures the timestamp difference between $\mathcal{B}_1^{p_h}$ and $\mathcal{B}_0^{p_h}$ is $1 \leq t_1^{p_a} - t_1^{p_h} < 9$ seconds (i.e., $\left\lfloor \frac{t_1^{p_a} - t_0^{p_h}}{9} \right\rfloor = 0$), making the difficulty of $\mathcal{B}_1^{p_a}$ higher than that of the honest block $\mathcal{B}_1^{p_h}$. According to Ethereum 1.x's fork selection rule, which prioritizes the chain with higher difficulty, other participants will prefer the adversary's block, ensuring attack success. Regardless of success, the process returns to the deployment phase to await the next valid timestamp difference. By precisely controlling timestamp differences, the RUM attack creates a sustained difficulty advantage without additional risk, exploiting the protocol's rules to secure preferential block inclusion and undermine blockchain fairness.

4. The Design of UUM Attack

After discussing the specific method of the RUM attack, we delve into the expansion of the strategy space in the UUM attack compared to the RUM attack, particularly during the deployment phase. The RUM attack imposes relatively strict constraints on the adversary's behavior, such as requiring specific conditions to enter the attack state, to ensure the risk-free feature of the attack. However, the UUM attack breaks this constraint, providing adversaries with more flexible strategic options.

To achieve minimal risk control, adversaries need to manipulate block timestamps with greater precision during the UUM attack process. By cleverly selecting timestamps, adversaries must not only ensure that their mined blocks have an advantage in the difficulty competition to increase the probability of attack success but also minimize the impact on the overall difficulty growth of the blockchain, making the attack more covert and sustainable. This requires adversaries to carefully consider the balance between various factors when executing the attack, aiming to reduce risks while obtaining benefits.

4.1. Method

Specifically, the UUM attack operationalizes a three phase framework to achieve low-risk, high-reward objectives by capitalizing on timestamp manipulation within Ethereum 1.x-style fork selection rules. In the deployment phase, the attacker monitors the timestamp difference between a newly generated honest block $\mathcal{B}_1^{p_h}$ and its preceding mainchain block $\mathcal{B}_0^{p_h}$, waiting for $t_1^{p_h} - t_0^{p_h} \in [9, +\infty)$ (i.e., $\left\lfloor \frac{t_1^{p_h} - t_0^{p_h}}{9} \right\rfloor \geq 1$), a condition that validates the honest block and enables selective timestamp manipulation to elevate the difficulty of the attacker's subsequent block above that of honest competitors. When the floor function result equals 1, the attack reduces to the risk-free RUM variant, whereas values greater than 1 introduce moderate risk alongside enhanced excess rewards.

Subsequently, in the execution phase, after satisfying deployment conditions, the attacker mines on the parent block of the latest mainchain block and, upon generating a valid block $\mathcal{B}_1^{p_a}$ before honest participants with a timestamp difference $t_a^{p_1} - t_a^{p_0} \in [1, 900)$, triggers successful attack progression to the risk control phase; this timestamp range ensures block validity and a difficulty advantage ($\mathcal{D}_1^{p_a} > \mathcal{D}_0^{p_h}$), thereby compelling honest nodes to prioritize the adversarial block during forks; failed conditions revert the process to deployment.

In the minimum risk control phase, the attacker meticulously adjusts the timestamp $t_a^{p_1}$ of their block to maximize difficulty superiority over honest blocks while minimizing subsequent difficulty growth rates through precise temporal calibration (e.g., enforcing $\left\lfloor \frac{t_1^{p_h} - t_0^{p_a}}{9} \right\rfloor = 1$ for minimal risk), after which the attack cycle resets to the deployment phase, establishing a self-reinforcing loop of strategic exploitation under controlled difficulty fluctuations.

Next, we elaborate on the relevant characteristics of the UUM attack under these more relaxed deployment conditions through Theorem 1, further understanding the advantages and impacts of the UUM attack compared to the RUM attack.

Theorem 1 (**UUM Initiation Condition**). *The initiation condition for the UUM attack is given by* $\left\lfloor \frac{t_1^{p_h} - t_0^{p_h}}{9} \right\rfloor \in [1, +\infty)$.

Proof of Theorem 1. The detailed proof of Theorem 1 can be found in Appendix B. □

The success of the UUM adversary's execution phase hinges on the combined effects of multiple critical factors, which not only encompass the difference in block timestamps but are also intimately related to the adversary's ability to successfully mine blocks that meet specific conditions. To gain an accurate understanding of the characteristics of the UUM attack during its execution phase, we precisely define the conditions for its success through Theorem 2.

Theorem 2 (**UUM Successful Condition**). *The UUM attack is successful if and only if* $\left\lfloor \frac{t_1^{p_h} - t_1^{p_a}}{9} \right\rfloor \in [1, +\infty)$ *and* $t_1^{p_a} - t_0^{p_a} \in [1, 900)$.

Proof of Theorem 2. The detailed proof of Theorem 2 can be found in Appendix C. □

In the course of our in-depth investigation of the UUM attack, we have already explored its initiation condition and success condition. However, for adversaries, minimizing the risks associated with the attack while pursuing success is also a crucial consideration. Here, risk primarily manifests in its impact on the difficulty of the blockchain.

Next, we will elaborate on the specific conditions for minimal risk control in the UUM attack through Theorem 3, further revealing the internal mechanisms and characteristics of the UUM attack under risk control strategies.

Theorem 3 (**UUM Successful Condition with Minimal Risk**). *The UUM attack is successful with minimal risk if and only if* $\left\lfloor \frac{t_1^{p_h} - t_1^{p_a}}{9} \right\rfloor = 1$ *and* $t_1^{p_a} - t_0^{p_a} \in [1, 900)$.

Proof of Theorem 3. The detailed proof of Theorem 3 can be found in Appendix D. □

As mentioned earlier, the RUM attack has stringent risk control requirements to achieve what is termed a risk-free attack. In contrast, the UUM attack relaxes these constraints to a certain extent, providing adversaries with a broader strategic space. Under specific settings of block timestamps, the UUM attack can exhibit similar or even equivalent characteristics to the RUM attack, providing an important perspective for us to deeply understand the feature of both attack methods.

Next, we will delve into the conditions under which the UUM attack downgrades into the RUM attack through Theorem 4.

Theorem 4 (**UUM Downgrades to RUM**). *The downgradation of UUM to RUM occurs if and only if* $\left\lfloor \frac{t_1^{p_h} - t_0^{p_h}}{9} \right\rfloor = 1$.

Proof of Theorem 4. The detailed proof of Theorem 4 can be found in E. □

4.2. State Space

Based on the analysis of the three phase strategy of the UUM attack, its state space can be divided into two categories: the deployment state and the attack state. The deployment state refers to the scenario where the UUM attacker waits for an appropriate opportunity to launch an attack. At this time, the blockchain topology and the timestamp of the latest block do not meet the attack conditions. The latest block is generated by either the attacker or honest participants, and the time difference between it and the previous main chain block is less than 9 seconds. The attack state means that the blockchain topology and the timestamp of the latest block comply with the attack conditions described in Theorem 2, that is, the latest block is generated by honest participants and the time difference between it and the previous main chain block is greater than or equal to 9 seconds.

Regarding state transitions, in the deployment state, if honest participants generate a block with a time difference greater than or equal to 9 seconds, the state changes to the attack state; otherwise, it remains in the deployment state. In the attack state, if honest participants continue to generate blocks with a time difference greater than or equal to 9 seconds, the attack state is maintained. If the attacker generates a block and strategically sets the timestamp, the state reverts to the deployment state. In other cases, the state also changes to the deployment state. The state transition process of UUM can be found in Figure 3-(b).

4.3. Reward Analysis

By cleverly manipulating block timestamps and employing attack strategies, UUM adversaries attempt to obtain rewards exceeding their fair share, which inevitably impacts the interests of honest participants. Next, we conduct an in-depth analysis through Theorem 5 to examine the changes in the share of coin-base rewards between adversaries and honest participants under the UUM attack, thereby revealing the specific impact of the UUM attack on the reward distribution mechanism.

Prior to proving this theorem, we define some additional notations: $\mathbb{R}_{\mathcal{P}}^{RUM}$ denotes the absolute share of the main chain block coin-base reward for an RUM participant $\mathcal{P}$, and $\mathbb{E}\left[\mathbb{R}_{\mathcal{P}}^{RUM}\right]$ represents the expected relative share of the main chain block coin-base reward for an RUM participant $\mathcal{P}$.

Theorem 5 (**UUM Adversary Rewards**). *The expected relative share of coin-base rewards for UUM adversaries increases, while the absolute share remains unchanged. Conversely, both the expected relative and absolute shares of coin-base rewards for honest participants will decrease.*

Proof of Theorem 5. Based on the analysis of state transitions for UUM deployment and attack states presented in Section 4.2, we calculate the absolute and relative shares of coin-base rewards for both UUM adversaries and honest participants.

The reward analysis for all state transitions is analogous to that of selfish mining. Among these, a particular case warrants attention, corresponding to the transition from the Attack State to the Deployment State (sixth row) in Table 1. In this case, the UUM adversary $\mathcal{P}_{\mathcal{A}}$ discovers the next valid block and publishes it, carefully setting the timestamp to ensure that the block's difficulty exceeds that of an honest block. This manipulation causes the attacker's block to be preferentially selected by other honest participants. Consequently, $\mathcal{P}_{\mathcal{A}}$ prevails in the fork competition and earns a coin-base reward. Conversely, the block generated by honest participants becomes an orphan block, triggering a transition from the Deployment State to the Attack State (third row) in Table 1. This transition results in the recall of a coin-base reward that was prematurely awarded to honest participants. Therefore, the reward $\mathcal{P}_{\mathcal{H}}$ for honest

participants corresponding to this state transition is -1. Details of the formalized proof are provided in Appendix F. □

The key mechanism lies in the irreversible shift of the steady-state probability:

- **Persistent Existence of the Attack State.** The deployment condition of the UUM attack ($\lfloor \frac{t_1^{p_h} - t_0^{p_h}}{9} \rfloor \geq 1$) is frequently triggered during the normal operation of the blockchain, causing the system to remain in the attack state for an extended period ($P(\text{Attack}) \uparrow$).
- **Self-Reinforcing Cycle.** When the attack is successful, the attacker suppresses the growth of subsequent block difficulty to the lowest level through timestamp calibration (see Theorem 3), making the attack cost approach zero ($AC^{\text{UUM}} \approx 0$). This enables the attacker to maintain the attack state indefinitely, forming a positive feedback loop of attack → reward extraction → controllable difficulty risk → continuous attack.
- **Zero-Sum Redistribution of Rewards.** Each time the attacker succeeds, the absolute reward share of honest participants decreases by 1 (as shown in the sixth row of State Transition Table II), while the relative share of the attacker grows linearly with the steady-state probability ($E[R_{P_A}^{\text{UUM}}] \propto P(\text{Attack})$). Since the difficulty adjustment cannot automatically correct this deviation, the reward distortion will be permanently embedded in the protocol's economic model.

5. The Design of SUUM Attack

The SUUM attack achieves persistent reward maximization through three core synergistic strategies embedded in its five phase framework, formally defined in Theorem 6-7 and visualized in Figure 3-(c). These strategies create a self-reinforcing loop by integrating timestamp manipulation, block withholding, and difficulty risk control:

- **Timestamp Manipulation for Difficulty Dominance.** During the Deployment Phase, once the adversary withholds the first private block $B_1^{p_a}$, timestamp manipulation becomes critical for difficulty advantage. By setting $t_1^{p_a} - t_0^{p_a} \in [1, 9)$ (Theorem 6 Condition 1), the attacker ensures: $\mathcal{D}_1^{p_a} = \mathcal{D}_0^{p_a} + \left(1 - \left\lfloor \frac{t_1^{p_a} - t_0^{p_a}}{9} \right\rfloor\right) \cdot \left\lfloor \frac{\mathcal{D}_0^{p_a}}{2048} \right\rfloor$, where $\left\lfloor \frac{t_1^{p_a} - t_0^{p_a}}{9} \right\rfloor = 0$ (due to $t < 9$), yielding a positive difficulty increment. In contrast, honest blocks with $t_1^{p_h} - t_0^{p_h} \geq 9$ incur a non-positive increment, ensuring $\mathcal{D}_1^{p_a} > \mathcal{D}_1^{p_h}$ and preferential chain selection (Figure 3-(c) transition from Attack State to Release Phase).
- **Block Withholding for Reorganization Cascades.** The withholding strategy extends to subsequent blocks ($i \geq 2$), where the adversary appends $B_i^{p_a}$ to the private chain until an honest block $B_i^{p_h}$ emerges. By controlling timestamp differences to satisfy $\left\lfloor \frac{t_i^{p_h} - t_{i-1}^{p_h} - (t_i^{p_a} - t_{i-1}^{p_a})}{9} \right\rfloor \geq 1$ (Theorem 6 Condition 2), each released private block surpasses the honest chain's difficulty, enabling cascading reorganizations. As shown in Figure 3-(c), the state transition from Attack i to Attack $i+1$ represents ongoing withholding, while Attack i to Deployment triggers strategic release, maximizing fork opportunities (simulations show SUUM forking rate 3.2% vs. UUM's 2.1% at $\alpha = 0.25$, Figure 6-(b)).
- **Difficulty Risk Control for Cost-free Persistence.** To minimize difficulty fluctuations, the adversary restricts $t_i^{p_a} - t_{i-1}^{p_a} \in [1, 9)$ (Theorem 7 Condition 1), ensuring $\left\lfloor \frac{t_i^{p_h} - t_i^{p_a}}{9} \right\rfloor = 1$ and minimal risk (maximal difficulty deviation 0.21, Figure 6-(a)). The state space's Downgrade State (Figure 3-(c)) acts as a safety mechanism, allowing fallback to UUM strategies when honest timestamp differences exceed 9 seconds, maintaining a steady-state attack probability 37.8% higher than UUM (Figure 4) and attack cost $\mathbb{AC}^{SUUM} \approx 0$ (Theorem 12).

5.1. Method

Specifically, the SUUM attack achieves the goal of low risk and high returns through five phases: SUUM downgrade phase, SUUM deployment phase, SUUM block withholding phase, SUUM block release phase, and SUUM minimum risk control phase.

- **Phase One: SUUM Downgrade Phase.** If adversary $\mathcal{P}_{\mathcal{A}}$ in SUUM observes that the difference between the timestamp $t_1^{p_h}$ of a newly generated block $\mathcal{B}_1^{p_h}$ by honest participant $\mathcal{P}_{\mathcal{H}}$ in the network and the timestamp $t_0^{p_h}$ of the previous main chain block $\mathcal{B}_0^{p_h}$ is $t_1^{p_h} - t_0^{p_h} \in [9, +\infty)$, i.e., $\left\lfloor \frac{t_1^{p_h} - t_0^{p_h}}{9} \right\rfloor \in [1, +\infty)$, then SUUM downgrades to UUM. Note that when $\left\lfloor \frac{t_1^{p_h} - t_0^{p_h}}{9} \right\rfloor = 1$, SUUM further downgrades to RUM. The subsequent strategies of adversary $\mathcal{P}_{\mathcal{A}}$ in SUUM are identical to those in UUM. If adversary $\mathcal{P}_{\mathcal{A}}$ in SUUM first discovers a valid block, it proceeds to **Phase Two**.
- **Phase Two: SUUM Deployment Phase.** At this point, adversary $\mathcal{P}_{\mathcal{A}}$ in SUUM first discovers a valid block $\mathcal{B}_1^{p_a}$. $\mathcal{P}_{\mathcal{A}}$ will withhold this block $\mathcal{B}_1^{p_a}$, forming a private chain visible only locally to himself, and then transitions to **Phase Three**.
- **Phase Three: SUUM Withholding Block Phase.** The adversary $\mathcal{P}_{\mathcal{A}}$ in SUUM continues mining along the private chain. If $\mathcal{P}_{\mathcal{A}}$ discovers a new valid block $\mathcal{B}_2^{p_a}$, he appends this block to its local private chain and continues executing **Phase Three**. If, on the other hand, an honest participant $\mathcal{P}_{\mathcal{H}}$ discovers a new valid block $\mathcal{B}_1^{p_h}$, it transitions to **Phase Four**.
- **Phase Four: SUUM Releasing Block Phase.** At this juncture, the local private chain of adversary $\mathcal{P}_{\mathcal{A}}$ in SUUM has a length of at least 1, and honest participant $\mathcal{P}_{\mathcal{H}}$ discovers a new valid block $\mathcal{B}_1^{p_h}$. Adversary $\mathcal{P}_{\mathcal{A}}$ in SUUM releases a private block $\mathcal{B}_1^{p_a}$ and transitions to **Phase Five**. After the process of releasing the block is completed, if the length of $\mathcal{P}_{\mathcal{A}}$'s local private chain still remains at least 1, he proceeds to **Phase Three**. If $\mathcal{P}_{\mathcal{A}}$

has no private blocks held in reserve, he transitions to **Phase One**.

- **Phase Five: SUUM Minimum Risk Control Phase.** $\mathcal{P}_{\mathcal{A}}$ carefully selects the timestamp $t_1^{p_a}$ for the block $\mathcal{B}_1^{p_a}$, aiming to achieve a higher difficulty $\mathcal{D}_1^{p_a}$ compared to the honest block's difficulty $\mathcal{D}_1^{p_h}$ while minimizing the subsequent block difficulty growth rate. Upon completion of this step, $\mathcal{P}_{\mathcal{A}}$ proceeds with the subsequent steps of **Phase Four**.

Compared to the UUM attack, the most notable distinction of the SUUM attack lies in its allowance for adversaries to strategically withhold or release blocks under specific circumstances. This characteristic enables adversaries to more flexibly respond to various situations within the blockchain network, increasing the likelihood of attack success and potential rewards. To fully grasp the mechanism of the SUUM attack, we first clarify its success conditions in Theorem 6.

Theorem 6 (**SUUM Successful Condition**). *The SUUM attack is successful if and only if the following conditions are met:*

(1) For $i = 1$*, the conditions* $\left\lceil \frac{t_i^{p_h} - t_{i-1}^{p_a}}{9} \right\rceil = \left\lceil \frac{t_1^{p_h} - t_0^{p_a}}{9} \right\rceil \in [1, +\infty)$ *and* $t_i^{p_a} - t_{i-1}^{p_a} = t_1^{p_a} - t_0^{p_a} \in [1, 900)$ *must be satisfied.*

(2) For $i \geq 2$*, the conditions* $\left\lceil \frac{t_i^{p_h} - t_{i-1}^{p_h} - \left(t_i^{p_a} - t_{i-1}^{p_a}\right)}{9} \right\rceil \in [1, +\infty)$ *and* $t_i^{p_a} - t_{i-1}^{p_a} \in [1, 900)$ *must be satisfied.*

Proof of Theorem 6. The detailed proof of Theorem 6 can be found in Appendix G. □

After thoroughly discussing the success conditions of the SUUM attack, we further focus on the minimum risk control strategy. Due to its more complex attack pattern involving operations such as withholding and releasing blocks, risk control becomes particularly important in the SUUM attack. Similar to the UUM attack, the primary risk associated with the SUUM attack manifests in its impact on blockchain difficulty.

Next, we will elaborate on the specific conditions of the SUUM attack in terms of minimal risk control through Theorem 7, revealing the internal mechanisms and characteristics of the SUUM attack under risk control strategies.

Theorem 7 (**SUUM Successful Condition with Minimal Risk**). *The SUUM attack is successful with minimal risk if and only if the following conditions are met:*

(1) For $i = 1$*, the conditions* $\left\lfloor \frac{t_i^{p_h} - t_{i-1}^{p_a}}{9} \right\rfloor = \left\lfloor \frac{t_1^{p_h} - t_0^{p_a}}{9} \right\rfloor = 1$ *and* $t_i^{p_a} - t_{i-1}^{p_a} = t_1^{p_a} - t_0^{p_a} \in [1, 900)$ *must be satisfied.*

(2) For $i \geq 2$*, the conditions* $\left\lfloor \frac{t_i^{p_h} - t_{i-1}^{p_h} - \left(t_i^{p_a} - t_{i-1}^{p_a}\right)}{9} \right\rfloor = 1$ *and* $t_i^{p_a} - t_{i-1}^{p_a} \in [1, 900)$ *must be satisfied.*

Proof of Theorem 7. The detailed proof of Theorem 7 can be found in Appendix H. □

5.2. State Space

Based on the analysis of the SUUM attack strategy, its state space is macroscopically divided into three states: the deployment state, the downgrade state, and the attack state, where the attack state is microscopically further divided into the withholding state and the releasing state. The deployment state refers to the scenario where the SUUM adversary waits for an opportune moment to launch an attack, where the blockchain topology and the timestamp of the latest block do not satisfy the attack conditions of the UUM adversary, and the latest block can be generated by honest participants (with previous states being the deployment, downgrade, or attack 1 state and a timestamp difference less than 9 seconds) or by the UUM adversary (with the previous state being the downgrade state). The downgrade state refers to the situation where the blockchain topology and the timestamp of the latest block meet the attack conditions of the UUM attack, prompting SUUM to degrade to UUM and execute its attack strategy. The attack state refers to the scenario where the blockchain topology and the timestamp of the latest block satisfy the SUUM attack conditions (i.e., the latest block is generated by honest participants with a timestamp difference of greater than or equal to 9 seconds from the previous mainchain block).

Regarding state transitions, the deployment state may transition to the downgrade state due to a timestamp difference of greater than or equal to 9 seconds in honest blocks, to the attack 1 state due to the adversary generating and withholding a block, or remain unchanged due to a timestamp difference of less than 9 seconds in honest blocks. The downgrade state may transition to the deployment state due to the adversary generating a block with strategic timestamp setting or a timestamp difference of less than 9 seconds in honest blocks, or remain in the downgrade state due to a timestamp difference of greater than or equal to 9 seconds in honest blocks. Within the attack state, regardless of the sub - state (withholding or releasing), if the next block is generated by honest participants, the adversary immediately releases the withheld block with strategic timestamp manipulation; if generated by the adversary, the block is continuously withheld. In both cases, the state ultimately transitions back to the deployment state. We show the state transition process of SUUM in Figure 3-(c).

5.3. Reward Analysis

After a detailed analysis of the mechanisms of the SUUM attack, including its success conditions and minimal risk success conditions, we focus our attention on the impact of the SUUM attack on the reward distribution pattern within blockchain networks. As we observed in our study of the UUM attack, any attack behavior has the potential to disrupt the originally fair reward distribution mechanism designed in blockchain systems, and the SUUM attack is no exception, with its impact being even more complex and far-reaching.

Theorem 8 (**SUUM Adversary Rewards**). *The expected relative share of the mainchain block coin-base reward for the SUUM adversary increases, while the absolute share remains unchanged. In contrast, both the expected relative share and the absolute share of the main chain block coin-base reward for honest participants will decrease.*

Proof of Theorem 8. Based on the analysis of state transitions in the Deployment State, Downgrade State, and Attack State of SUUM presented in Section 5.2, we calculate the absolute and relative shares of coin-base rewards for both the SUUM adversary and honest participants.

The reward analysis for all state transitions is analogous to that of the UUM attack. Three special cases deserve attention, corresponding respectively to the transitions from the Downgrade state to the Deployment state (sixth row) in Table 2, from the Attack 1 state to the Deployment state (seventh row), and from the Attack $i,\ i \geq 1$ state to the Attack $i+1$ state (eighth row).

In the first case, the SUUM adversary $\mathcal{P}_{\mathcal{A}}$, while in the Downgrade state, discovers the next valid block and publishes it. By carefully setting the timestamp, $\mathcal{P}_{\mathcal{A}}$ ensures that the block's difficulty exceeds that of honest blocks, causing the attacker's block to be preferentially selected by other honest participants. Consequently, $\mathcal{P}_{\mathcal{A}}$ wins the fork competition and earns a coin-base reward. The block generated by the honest participant coalition becomes an orphan block, prompting a transition to the Attack state (third row) and necessitating the recall of a pre-paid coin-base reward intended for honest participants. Therefore, the reward $\mathcal{P}_{\mathcal{H}}$ for honest participants corresponding to this state transition is -1.

In the second case, an honest participant discovers and publishes a valid block while in the Attack 1 state. Immediately, the SUUM adversary $\mathcal{P}_{\mathcal{A}}$ releases a withheld block, carefully setting its timestamp to ensure its difficulty exceeds that of the honest block. Ultimately, the selfish player wins the fork competition, rendering the honest block an orphan, and $\mathcal{P}_{\mathcal{A}}$ earns a coin-base reward (pre-paid during the transition from the Deployment state to the Attack 1 state in the first row). The honest player receives no reward and triggers a transition from the Attack 1 state to the Deployment state.

In the third case, when the SUUM adversary $\mathcal{P}_{\mathcal{A}}$ is in the Attack $i,\ i \geq 1$ state and discovers the next valid block, $\mathcal{P}_{\mathcal{A}}$ withholds it and transitions the current state to the next state, Attack $i+1$. Regardless of subsequent blockchain evolutions, by meticulously setting the timestamp, $\mathcal{P}_{\mathcal{A}}$ ensures that the withheld block ultimately becomes part of the main chain. Consequently, $\mathcal{P}_{\mathcal{A}}$ earns a coin-base reward while causing a corresponding loss of a coin-base reward for competing honest participants.

The detailed proof of Theorem 8 can be found in Appendix I. □

6. Comparison

Compared to the RUM attack, the UUM attack significantly expands the strategic space. The RUM attack emphasize risklessness, with relatively strict attack conditions, whereas the UUM attack allows adversaries to obtain higher potential rewards while bearing a certain level of risk. This change is primarily reflected in the relaxation of conditions during the deployment and execution stages of the UUM attack, enabling adversaries to initiate attacks under a wider range of network states. From the perspective of reward distribution, the UUM attack results in an increase in the expected relative share of adversaries and a decrease in the share of honest participants, disrupting the original reward balance based on fair computing power input.

The SUUM attack further deepens the strategic capabilities of adversaries based on the UUM attack. They introduce strategies for withholding and releasing blocks, enabling adversaries to more flexibly respond to dynamic changes in blockchain networks. In terms of success conditions and minimal risk success conditions, the SUUM attack involves more complex block timestamp relationships and phase judgments, reflecting the high complexity of their attack strategies. Similar to the UUM attack, the SUUM attack also alters the reward distribution pattern, benefiting adversaries while damaging honest participants. However, due to its more powerful strategic space, the SUUM attack has a more profound impact on blockchain networks. Next, we detailedly compare the advantages of these attack methods in terms of rewards in Theorem 9.

Theorem 9 (**Reward Comparison of Uncle Maker-based Attacks and Honest Mining**). *SUUM outperforms UUM, which in turn outperforms RUM, all of which surpass honest mining.*

Proof of Theorem 9. The detailed proof of Theorem 9 can be found in Appendix J. □

The RUM attack maintains a probabilistically negligible difficulty risk due to its constrained operation, which inherently limits both its attack surface and potential impact on blockchain difficulty adjustment. In contrast, the UUM attack's relaxation of temporal constraints introduces measurable but bounded difficulty risk, primarily determined by the Markovian state transition probabilities post-attack. The SUUM attack exhibits the highest difficulty risk profile, as its compounded strategy of staggered block release and timestamp manipulation in both attack and downgrade states creates non-linear interactions with the difficulty adjustment algorithm. Next, we use Theorem 10 to elaborate on the differences in blockchain difficulty risk between these attack methods and honest mining.

Theorem 10 (**Difficulty Risk Comparison of Uncle Maker-based Attacks and Honest Mining**). *Compared to honest mining, the difficulty risks posed by RUM, UUM, and SUUM attacks on blockchain difficulty are as follows:*

(1) The increased difficulty risk associated with the RUM attack is denoted by $\sum \left(Attack \overset{\mathcal{P}_{\mathcal{A}}}{\Rightarrow} \overset{any\ t}{} Deploy \right)$*, which represents the sum of the instances where a RUM adversary successfully attacks and drives the Attack State to transition to the Deployment State..*

(2) *The increased difficulty risk associated with the UUM attack is denoted by* $\sum \left(Attack \overset{\mathcal{P}_{\mathcal{A}}}{\Rightarrow}{}^{any\ t} Deploy \right)$, *which represents the sum of the instances where a UUM adversary successfully attacks and drives the Attack State to transition to the Deployment State.*

(3) *The increased difficulty risk associated with SUUM is denoted by* $\sum \begin{pmatrix} Downgrade \overset{\mathcal{P}_{\mathcal{A}}}{\Rightarrow}{}^{any\ t} Deploy \\ +Attack\ 1 \overset{\mathcal{P}_{\mathcal{H}}}{\Rightarrow}{}^{any\ t} Deploy \end{pmatrix}$, *which represents the sum of the instances of two types of state transitions corresponding to successful attacks by an SUUM adversary.*

Proof of Theorem 10. The detailed proof of Theorem 10 can be found in Appendix K. □

Through the preceding theoretical analysis of three types of Uncle Maker-based attacks, we understand that their essence lies in maliciously creating uncle blocks to induce forks in the blockchain and meticulously manipulating timestamps to prioritize the adoption of adversary blocks by other network participants. Consequently, the magnitude of the deliberate forking rate serves as a metric for assessing the severity of Uncle Maker-based attacks. Theorems 9 and 10 compare RUM, UUM, SUUM, and honest mining from the perspectives of reward and risk, respectively. Intuitively, as the maliciousness of RUM, UUM, and SUUM increases, their forking rates should also increase correspondingly. To streamline our notation, we use $\mathbb{P}_S^A$ to denote the steady-state probability of state S when adversary $\mathcal{A}$ employs attack A, and $\mathbb{FR}^A$ to represent the forking rate induced by attack A. In the following, we formally present a comparison of RUM, UUM, SUUM, and honest mining in terms of block forking rates.

Theorem 11 (Forking Rate Comparison of Uncle Maker-based Attacks and Honest Mining). *For an adversary $\mathcal{A}$ with the same computational power, the block forking rates induced by adopting RUM, UUM, SUUM, and honest mining satisfy the following relationship:*

$$\mathbb{FR}^{SUUM} > \mathbb{FR}^{UUM} > \mathbb{FR}^{RUM} > \mathbb{FR}^{HM}. \tag{1}$$

Specifically,

(1) *For honest mining,* $\mathbb{FR}^{HM} = 0$.

(2) *For RUM,* $\mathbb{FR}^{RUM} = \mathbb{P}^{RUM}_{Attack} \cdot \mathcal{P}_{\mathcal{A}}(t < 9)$, *where* $\mathcal{P}_{\mathcal{A}}(t < 9)$ *represents a scenario where an adversary $\mathcal{A}$ in RUM attack state finds the next block, and the difference in timestamps between it and its parent block is less than* 9.

(3) *For UUM,* $\mathbb{FR}^{UUM} = \mathbb{P}^{UUM}_{Attack} \cdot \mathcal{P}_{\mathcal{A}}(any\ t)$, *which indicates a scenario where an adversary $\mathcal{A}$ in the UUM attack state finds the next block, regardless of the difference in timestamps between it and its parent block.*

(4) *For SUUM,* $\mathbb{FR}^{SUUM} = \mathbb{P}^{SUUM}_{Attack} + \mathbb{P}^{SUUM}_{Downgrade} \cdot \mathcal{P}_{\mathcal{A}}(any\ t)$, *which represents the sum of the steady-state probability of being in the attack state and the probability of an adversary in the downgrade state finding the next block, regardless of the difference in timestamps between it and its parent block.*

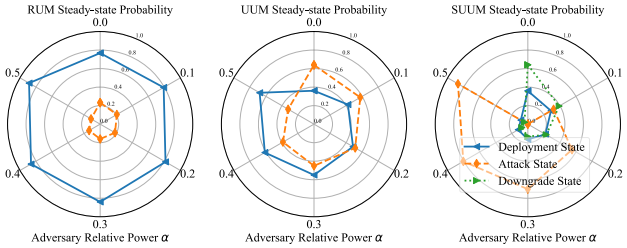


Figure 4. Steady-state Probability. This figure illustrates the steady-state probabilities of three different attack strategies as a function of the adversary's relative power α. Note that we only show the total probability of all Attack states within the SUUM model.

Proof of Theorem 11. The detailed proof of Theorem 11 can be found in Appendix L. □

Next, we focus our attention on the attack cost associated with three types of attacks. It is noteworthy that we characterize the attack cost as the reduction in the adversary's block generation probability resulting from an increase in block difficulty. Prior to the detailed analysis, we first introduce some additional notations. We denote $\mu_{\mathcal{A}}$ as the proportion of power controlled by adversary $\mathcal{A}$, max_target as the maximum target value, and $\mathbb{AC}$ as the attack cost.

Subsequently, in Theorem 12, we formally present the attack cost for these three types of attacks.

Theorem 12 (Cost Comparison of Uncle Maker-based Attacks). *For an adversary $\mathcal{A}$ possessing the same relative power, the attack cost of RUM, UUM, and SUUM satisfy the following relationship:*

$$\mathbb{AC}^{SUUM} = \mathbb{AC}^{UUM} \approx \mathbb{AC}^{RUM} = 0. \tag{2}$$

Specifically,

(1) *For RUM,* $\mathbb{AC}^{RUM} = 0$.

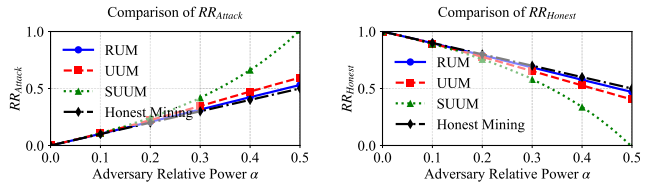
(2) *For UUM or SUUM,* $\mathbb{AC}^{UUM} = \mathbb{AC}^{SUUM} = \mu_{\mathcal{A}} \cdot \frac{\mathcal{D}_0^{Ph} - \mathcal{D}_1^{Ph}}{max\_target} \leq \mu_{\mathcal{A}} \cdot \frac{1}{2^{216.35}} \approx 0$.

Proof. The detailed proof of Theorem 12 can be found in Appendix M. □

This result once again implies that attackers can relatively easily carry out UUM and SUUM attacks with almost no cost risk, thereby further highlighting the severe threats posed by these two attack strategies to the security of timestamp-based Nakamoto-style blockchains. Since nearly cost-free attacks may prompt more attackers to attempt to exploit these vulnerabilities, undermining the fairness and stability of the blockchain.

7. Simulated Estimation

To empirically validate the theoretical analysis of the UUM and SUUM attacks, we conducted extensive simulations under various adversarial power ratios. These simulations aimed to quantify the steady-state probabilities, reward distributions, difficulty risks, and forking rates associated



(a) RR_{Attack} (b) RR_{Honest}

Figure 5. Comparison of Relative Rewards under different Mining Strategies. (a) Comparison of Adversary Relative Rewards under Different Mining Strategies. This figure compares the relative reward gains of adversaries under different mining strategies, illustrating the reward disparities among the SUUM, UUM, and RUM attack strategies compared to honest mining. The results demonstrate that SUUM yields the highest rewards, significantly outperforming UUM and RUM, while all three attack strategies surpass honest mining in profitability. This outcome confirms that adversaries can obtain excess rewards by manipulating timestamps and strategically withholding blocks. (b) Comparison of Honest Participant Relative Rewards under Different Mining Strategies. This figure presents a comparative analysis of honest participants' relative rewards under different mining strategies. Notably, SUUM exhibits the most severe reward suppression effect, followed by UUM and RUM, with all three attack strategies significantly undercutting the baseline rewards achievable through honest mining. These results quantitatively validate how timestamp manipulation and strategic block withholding by adversaries systematically disadvantage honest participants.

with each attack strategy. By comparing these metrics with honest mining and the baseline RUM attack, we demonstrate the enhanced profitability and persistence of the proposed advanced variants. The simulation setup and results are detailed in the following subsections.

We implemented a discrete-event simulator to model the Ethereum 1.x blockchain protocol, incorporating the fork selection rules and difficulty adjustment mechanisms described in Section 3. The simulator tracks block generation, timestamp manipulation, and adversarial strategies under controlled conditions. Key parameters include:

- **Adversarial power ratio α:** Varied from 0 to 0.5 in increments of 0.05.
- **Block generation:** Modeled as a Poisson process with a 13-second average block time.
- **Timestamp constraints:** Enforced Ethereum 1.x style blockchain validity rules.
- **Difficulty adjustment:** Calculated dynamically based on block timestamps and parent difficulties.

Each simulation ran for 1,000,000 blocks to ensure convergence to steady-state behavior, with results averaged over 10,000 trials to minimize variance. The following subsections present the findings for each evaluated metric.

7.1. Estimate the Steady-state Probability

To quantitatively analyze the effectiveness of the proposed attack strategies, we estimate the steady-state probabilities of the RUM, UUM, and SUUM attacks under varying adversarial power ratios. The steady-state probability reflects the long-term likelihood of the system being in an attack state, which directly correlates with the adversary's ability to sustain the attack and maximize rewards.

As shown in Figure 4, the steady-state probability of the deployment state increases with the adversary's power for RUM and UUM strategies. However, SUUM exhibits the highest steady-state probability of the deployment state across higher power levels, followed by UUM and RUM. This is because SUUM's ability to withhold and strategically release blocks expands its attack opportunities, while UUM's relaxation of risk-free constraints allows more frequent transitions to the attack state compared to RUM.

The results validate that SUUM's design achieves superior persistence in maintaining the attack state, enabling adversaries to exert sustained influence on the blockchain. This aligns with Theorem 7, where SUUM's reward advantage stems from its higher steady-state probability of attack execution. The findings underscore the need for countermeasures to mitigate such persistent attacks, as discussed in Section 8.

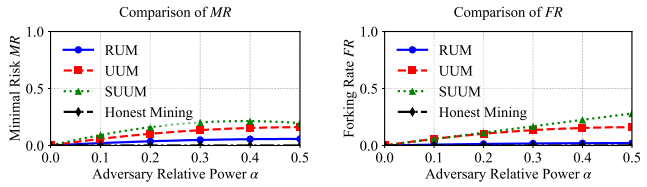
7.2. Estimate the Relative Reward

To evaluate the economic impact of the proposed attacks, we analyze the relative rewards obtained by adversaries and honest participants under RUM, UUM, SUUM, and honest mining strategies. The relative reward measures the proportion of total block rewards captured by each party, reflecting the fairness and security of the blockchain's incentive mechanism.

The relative rewards for adversaries (RR_{Attack}) and honest participants (RR_{Honest}) are calculated by dividing their earned rewards by the total system rewards. As shown in Figure 5, SUUM consistently yields the highest relative rewards for adversaries, surpassing UUM and RUM across all power levels. For instance, at $\alpha = 0.25$, SUUM enables adversaries to capture 33.30% of the rewards, while UUM and RUM achieve 28.41% and 26.12%, respectively. This aligns with Theorem 9, where SUUM's block withholding and timestamp manipulation synergistically amplify rewards. Honest mining, as expected, adheres to the fair share, serving as the baseline.

Figure 5 demonstrates the corresponding degradation in honest participants' rewards under adversarial strategies. SUUM inflicts the most severe suppression, reducing honest rewards by up to 11.85% compared to honest mining at $\alpha = 0.3$. UUM and RUM exhibit intermediate effects, with honest rewards declining linearly as α increases. This inverse relationship between adversarial and honest rewards confirms the zero-sum nature of reward redistribution in these attacks.

The death spiral effect of SUUM, emerges from a self-reinforcing cycle where adversarial rewards come at the direct expense of honest participants, driving a catastrophic breakdown of the protocol's economic model.



(a) MR (b) FR

Figure 6. Comparison of Minimal Difficulty Risk and Forking Rates under Different Attack Strategies. (a) Comparison of Minimal Difficulty Risk under Different Attack Strategies. This figure presents a comparative assessment of the minimal difficulty risk levels associated with different attack strategies. Honest mining maintains a baseline risk level of zero, while RUM introduces only minimal difficulty escalation risk. In contrast, UUM exhibits marginally higher risk due to its less restrictive attack conditions, and SUUM demonstrates the most significant risk elevation attributable to its block withholding mechanism. (b) Comparison of Forking Rates under Different Mining Strategies. This figure presents a comparative analysis of forking rates under different attack strategies, illustrating the relationship between the adversary's relative power and the resultant forking rate. The results demonstrate that honest mining maintains a zero forking rate due to strict protocol compliance, while the RUM, UUM, and SUUM attacks exhibit progressively increasing fork probabilities. This trend originates from fundamental differences in attack mechanisms.

7.3. Estimate the Minimal Difficulty Risk

The results underscore how SUUM's advanced strategies exploit the fork selection rule more effectively than UUM or RUM. The disproportionate rewards stem from two factors:

- **Timestamp Manipulation:** Adversaries artificially inflate their block difficulty (Section 4.1), increasing mainchain inclusion.
- **Block Withholding:** Delayed releases (Section 5.1) disrupt honest miners' progress, compounding rewards over time.

This analysis quantifies the economic incentives driving Uncle Maker attacks, emphasizing the urgency of mitigations (Section 8) to restore reward fairness. The significant reward gaps shown in Figure 5 further motivate our proposed countermeasures, such as timestamp verification and difficulty algorithm adjustments.

A critical aspect of Uncle Maker attacks is their impact on blockchain difficulty, which influences long-term network stability. Here, we evaluate the minimal difficulty risk-the least additional risk imposed on the blockchain's difficulty adjustment mechanism by each attack strategy. This metric reflects how subtly adversaries can execute attacks without destabilizing the network.

The minimal difficulty risk is quantified as $MR = \mathcal{D}_{Attack} - \mathcal{D}_{Honest}$, where $\mathcal{D}_{Attack}$ and $\mathcal{D}_{Honest}$ represent the difficulty of adversarial and honest blocks at the same highest height. Results are averaged across 10,000 trials to ensure statistical robustness.

Figure 6 reveals distinct risk profiles for each strategy. Honest Mining maintains a baseline risk of zero, as no difficulty manipulation occurs. RUM introduces negligible risk, as its strict timestamp constraints (Theorem 3) limit difficulty fluctuations. UUM exhibits marginally higher risk due to relaxed initiation conditions (Theorem 1), allowing occasional difficulty spikes. SUUM poses the highest risk, as its block withholding strategy (Section 5.1) disrupts difficulty adjustment continuity. We further heuristically find three observations:

- **Low Difficulty Risk Attacks:** All strategies induce sub-0.21 difficulty risk, confirming that Uncle Maker attacks are low difficulty risk.
- **Trade-off with Profitability:** SUUM's higher risk aligns with its superior rewards (Section 7), demonstrating a risk-reward balance.
- **Network Stability:** Even SUUM's maximal risk remains manageable (e.g., < 0.21 at $\alpha = 0.37$), explaining why such attacks could persist undetected in practice.

Crucially, SUUM's minimal difficulty risk allows this death spiral to proceed undetected, as the attack remains cost-free (Theorem 12). By calibrating timestamps to 1-second granularity (Section 5.1), adversaries avoid triggering difficulty spikes that would otherwise penalize their mining efficiency. This stealth enables sustained reward extraction, as seen in Figure 5-(b). The result is a low-risk, high-reward environment that incentivizes even moderate power miners to defect, amplifying the spiral.

7.4. Estimate the Forking Rate

SUUM's elevated forking rate directly correlates with honest reward suppression in Figure 5-(b). Each fork invalidates honest blocks, reducing their effective hash power contribution and further discouraging participation. This creates a vicious cycle: higher forking $\rightarrow$ fewer honest blocks $\rightarrow$ lower incentives $\rightarrow$ more miners abandon honesty, as reflected in the steep decline of honest relative rewards for SUUM compared to UUM/RUM.

We simulate RUM, UUM, SUUM, and honest mining over 1,000,000 blocks, measuring the forking rate (FR), where the results are averaged across 10,000 trials, with adversarial power α varying from 0 to 0.5. We note that the forking rate is defined as the number of intentionally induced forks divided by the total number of blocks mined in the system.

Figure 6 highlights stark contrasts in forking behavior. Honest Mining maintains a 0% forking rate, as all participants adhere to the canonical chain. RUM triggers occasional forks, occurring only when adversaries mine blocks with timestamps $t < 9$ (Theorem 11). UUM exhibits higher fork rates, as its relaxed constraints (Section 4.1) permit more frequent adversarial interventions. SUUM maximizes forks by combining withheld block releases (Section 5.1) and timestamp manipulation, amplifying chain reorganizations.

The death spiral effec underscores SUUM's unique threat: unlike prior attacks, it does not merely increase adversarial rewards but systematically erodes the incentive for honest participation, leading to irreversible protocol collapse. The combination of cost-free persistence (Theorem 12), recursive timestamp manipulation, and cascading reorganizations creates a feedback loop that accelerates as

more miners defect. This highlights the urgent need for mitigations that break this cycle (Section 8), to restore honest miners' incentives and prevent systemic failure.

8. Discussion

We now analyze the essence of the Uncle Maker attack and propose some mitigation measures to effectively prevent the Uncle Maker Attack and Its Advanced Variants. These measures mainly focus on adjusting the consensus mechanism and incentive mechanism of the blockchain, aiming to reduce the profit space of adversaries and enhance the security and fairness of the blockchain system. The details can be found in Appendix N.

9. Conclusion

This study demonstrates that timestamp-based Nakamoto-style blockchains, particularly Ethereum 1.x derivatives, face existential threats from the SUUM attack. Unlike transient adversarial strategies, SUUM inflicts permanent systemic damage by irreversibly distorting the protocol's incentive structure through three synergistic mechanisms: (1) precision timestamp manipulation to inflate adversarial difficulty advantages, (2) strategic block withholding to amplify chain reorganizations, and (3) granular difficulty risk control to ensure cost-free sustainability. These mechanisms create a self-reinforcing cycle: adversarial rewards scale super-linearly (e.g., achieving 43.7% rewards at $\alpha = 0.4$), while honest participants face diminishing returns (11.85% loss at $\alpha = 0.3$), ultimately incentivizing rational miners to defect and accelerating protocol collapse.

Our large-scale simulations (1M blocks, 10K trials) validate SUUM's dominance over prior attacks. SUUM adversaries achieve a 33.30% reward share at $\alpha = 0.25$, surpassing UUM (28.41%) and RUM (26.12%), while inducing higher forking rates (3.2% vs. UUM's 2.1% and RUM's 1.4%). Crucially, SUUM bypasses traditional constraints like hash power thresholds by exploiting timestamp calibration, maintaining minimal difficulty escalation (maximal risk: 0.21) despite sustained exploitation. This asymmetric advantage stems from the protocol's inability to self-correct reward distortions, leading to a death spiral where adversarial coalitions dominate and honest participation becomes economically untenable.

Future work must prioritize protocol redesigns that decouple difficulty adjustments from adversarial timestamp control and introduce economic penalties for abnormal block intervals. This work underscores the urgent need to rethink consensus layer security assumptions in timestamp-dependent systems, balancing incentive compatibility with adversarial resilience in next generation blockchain designs.

References

[1] C. Dwork and M. Naor, "Pricing via processing or combatting junk mail," in *Proceedings of the 12th Annual International Cryptology Conference on Advances in Cryptology*, ser. CRYPTO '92. Berlin, Heidelberg: Springer-Verlag, 1992, p. 139–147.

[2] M. Jakobsson and A. Juels, "Proofs of work and bread pudding protocols," in *Proceedings of the IFIP TC6/TC11 Joint Working Conference on Secure Information Networks: Communications and Multimedia Security*, ser. CMS '99. NLD: Kluwer, B.V., 1999, p. 258–272.

[3] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2009.

[4] V. Buterin, "Ethereum whitepaper," 2024, https://ethereum.org/whitepaper/.

[5] ——, "A next-generation smart contract and decentralized application platform," 2014.

[6] coinmarketcap.com, "Cryptocurrency market capitalizations," (2024), https://coinmarketcap.com/.

[7] R. Zhang and B. Preneel, "Lay down the common metrics: Evaluating proof-of-work consensus protocols' security," in *2019 IEEE Symposium on Security and Privacy (SP)*, 2019, pp. 175–192.

[8] Y. Lewenberg, Y. Sompolinsky, and A. Zohar, "Inclusive block chain protocols," in *Financial Cryptography and Data Security: 19th International Conference, FC 2015, San Juan, Puerto Rico, January 26-30, 2015, Revised Selected Papers 19*. Berlin, Heidelberg: Springer, 2015, pp. 528–547, https://doi.org/10.1007/978-3-662-47854-7_33.

[9] R. Pass and E. Shi, "Fruitchains: A fair blockchain," in *Proceedings of the ACM Symposium on Principles of Distributed Computing*, ser. PODC '17. New York, NY, USA: Association for Computing Machinery, 2017, p. 315–324. [Online]. Available: https://doi.org/10.1145/3087801.3087809

[10] R. Pass and E. Shi, "Hybrid Consensus: Efficient Consensus in the Permissionless Model," in *31st International Symposium on Distributed Computing (DISC 2017)*, ser. Leibniz International Proceedings in Informatics (LIPIcs), A. Richa, Ed., vol. 91. Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2017, pp. 39:1–39:16. [Online]. Available: https://drops-dev.dagstuhl.de/entities/document/10.4230/LIPIcs.DISC.2017.39

[11] I. Eyal, A. E. Gencer, E. G. Sirer, and R. Van Renesse, "Bitcoin-ng: a scalable blockchain protocol," in *Proceedings of the 13th Usenix Conference on Networked Systems Design and Implementation*, ser. NSDI'16. USA: USENIX Association, 2016, p. 45–59.

[12] J. Wang and H. Wang, "Monoxide: Scale out blockchains with asynchronous consensus zones," in *16th USENIX Symposium on Networked Systems Design and Implementation (NSDI 19)*. Boston, MA: USENIX Association, Feb. 2019, pp. 95–112. [Online]. Available: https://www.usenix.org/conference/nsdi19/presentation/wang-jiaping

[13] J. Katz and Y. Lindell, *Introduction to Modern Cryptography, Second Edition*, 2nd ed. Chapman & Hall/CRC, 2014.

[14] A. E. K. R. Bowden, H. P. Keeler and P. G. Taylor, "Modeling and analysis of block arrival times in the bitcoin blockchain," *Stochastic Models*, vol. 36, no. 4, pp. 602–637, 2020. [Online]. Available: https://doi.org/10.1080/15326349.2020.1786404

[15] Y. Sompolinsky and A. Zohar, "Secure high-rate transaction processing in bitcoin," in *Financial Cryptography and Data Security*, R. Böhme and T. Okamoto, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015, pp. 507–527.

[16] A. Yaish, S. Tochner, and A. Zohar, "Blockchain stretching & squeezing: Manipulating time for your best interest," in *Proceedings of the 23rd ACM Conference on Economics and Computation*, ser. EC '22. New York, NY, USA: Association for Computing Machinery, 2022, p. 65–88. [Online]. Available: https://doi.org/10.1145/3490486.3538250

[17] C. Li, P. Li, D. Zhou, Z. Yang, M. Wu, G. Yang, W. Xu, F. Long, and A. C.-C. Yao, "A decentralized blockchain with high throughput and fast confirmation," in *Proceedings of the 2020 USENIX Conference on Usenix Annual Technical Conference*, ser. USENIX ATC'20. USA: USENIX Association, 2020.

[18] C. Li, P. Li, D. Zhou, W. Xu, F. Long, and A. Yao, "Scaling nakamoto consensus to thousands of transactions per second," 2018. [Online]. Available: https://arxiv.org/abs/1805.03870

[19] H. Yu, I. Nikolić, R. Hou, and P. Saxena, "Ohie: Blockchain scaling made simple," in *2020 IEEE Symposium on Security and Privacy (SP)*, 2020, pp. 90–105.

[20] Y. Sompolinsky, S. Wyborski, and A. Zohar, "Phantom ghostdag: a scalable generalization of nakamoto consensus," in *Proceedings of the 3rd ACM Conference on Advances in Financial Technologies*, ser. AFT '21. New York, NY, USA: Association for Computing Machinery, 2021, p. 57–70. [Online]. Available: https://doi.org/10.1145/3479722.3480990

[21] E. PoW, "Ethereum pow," (2024), https://ethereumpow.org/.

[22] E. Fair, "Ethereum fair," (2024), https://etherfair.org/.

[23] E. Classic, "Ethereum classic," (2024), https://ethereumclassic.org/.

[24] Miningpoolstats, "Mining pool stats," (2024), https://miningpoolstats.stream/.

[25] S. Fork, "Fork," 2024. [Online]. Available: https://en.wikipedia.org/wiki/Fork_(blockchain)

[26] D. Meshkov, A. Chepurnoy, and M. Jansen, "Short paper: Revisiting difficulty control for blockchain systems," in *Data Privacy Management, Cryptocurrencies and Blockchain Technology*, J. Garcia-Alfaro, G. Navarro-Arribas, H. Hartenstein, and J. Herrera-Joancomartí, Eds. Cham: Springer International Publishing, 2017, pp. 429–436.

[27] P. Katsiampa, S. Corbet, and B. Lucey, "High frequency volatility co-movements in cryptocurrency markets," *Journal of International Financial Markets, Institutions and Money*, vol. 62, pp. 35–52, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S104244311930023X

[28] A. Yaish and A. Zohar, "Correct Cryptocurrency ASIC Pricing: Are Miners Overpaying?" in *5th Conference on Advances in Financial Technologies (AFT 2023)*, ser. Leibniz International Proceedings in Informatics (LIPIcs), J. Bonneau and S. M. Weinberg, Eds., vol. 282. Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023, pp. 2:1–2:25. [Online]. Available: https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.AFT.2023.2

[29] G. Goren and A. Spiegelman, "Mind the mining," in *Proceedings of the 2019 ACM Conference on Economics and Computation*, ser. EC '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 475–487. [Online]. Available: https://doi.org/10.1145/3328526.3329566

[30] A. Fiat, A. Karlin, E. Koutsoupias, and C. Papadimitriou, "Energy equilibria in proof-of-work mining," in *Proceedings of the 2019 ACM Conference on Economics and Computation*, ser. EC '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 489–502. [Online]. Available: https://doi.org/10.1145/3328526.3329630

[31] D. I. Ilie, S. M. Werner, I. D. Stewart, and W. J. Knottenbelt, "Unstable throughput: When the difficulty algorithm breaks," in *2021 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*, 2021, pp. 1–5.

[32] satofishi, "I can't stop appreciate this elegant implementation of whatwe've done over the past two years," 2024. [Online]. Available: https://twitter.com/satofishi/status/1556518282383036416

[33] ——, "We respect the consensus as is," 2024. [Online]. Available: https://twitter.com/satofishi/status/1556510404116889600

[34] I. Eyal and E. G. Sirer, "Majority is not enough: bitcoin mining is vulnerable," *Commun. ACM*, vol. 61, no. 7, p. 95–102, Jun. 2018. [Online]. Available: https://doi.org/10.1145/3212998

[35] BitInfoCharts, "Bitcoin, ethereum, dogecoin, xrp, ethereum clas sic, litecoin, monero, bitcoin cash, zcash, bitcoin gold hashrate historical chart." 2024. [Online]. Available: https://web.archive.org/web/20220522122528/https://bitinfocharts.com/comparison/hashrate-btc-eth-doge-xrp-etc-ltc-xmr-bch-zec-btg.html#3y)

[36] A. Gervais, G. O. Karame, K. Wüst, V. Glykantzis, H. Ritzdorf, and S. Capkun, "On the security and performance of proof of work blockchains," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '16. New York, NY, USA: Association for Computing Machinery, 2016, p. 3–16. [Online]. Available: https://doi.org/10.1145/2976749.2978341

[37] S. Azouvi and A. Hicks, "Sok: Tools for Game Theoretic Models of Security for Cryptocurrencies," *Cryptoeconomic Systems*, vol. 0, no. 1, apr 5 2021, https://cryptoeconomicsystems.pubpub.org/pub/azouvi-sok-security.

[38] J. Bonneau, A. Miller, J. Clark, A. Narayanan, J. A. Kroll, and E. W. Felten, "Sok: Research perspectives and challenges for bitcoin and cryptocurrencies," in *2015 IEEE Symposium on Security and Privacy*, 2015, pp. 104–121.

[39] A. Yaish, G. Stern, and A. Zohar, "Uncle maker:(time) stamping out the competition in ethereum," in *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*, 2023, pp. 135–149.

[40] A. Sapirshtein, Y. Sompolinsky, and A. Zohar, "Optimal selfish mining strategies in bitcoin," in *Financial Cryptography and Data Security*, J. Grossklags and B. Preneel, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2017, pp. 515–532.

[41] Z. Wang, J. Liu, Q. Wu, Y. Zhang, H. Yu, and Z. Zhou, "An analytic evaluation for the impact of uncle blocks by selfish and stubborn mining in an imperfect ethereum network," *Comput. Secur.*, vol. 87, no. C, Nov. 2019. [Online]. Available: https://doi.org/10.1016/j.cose.2019.101581

[42] I. Eyal, "The miner's dilemma," in *Proceedings of the 2015 IEEE Symposium on Security and Privacy*, ser. SP '15. USA: IEEE Computer Society, 2015, p. 89–103. [Online]. Available: https://doi.org/10.1109/SP.2015.13

[43] Y. Kwon, D. Kim, Y. Son, E. Vasserman, and Y. Kim, "Be selfish and avoid dilemmas: Fork after withholding (faw) attacks on bitcoin," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '17. New York, NY, USA: Association for Computing Machinery, 2017, p. 195–209. [Online]. Available: https://doi.org/10.1145/3133956.3134019

[44] N. T. Courtois and L. Bahack, "On subversive miner strategies and block withholding attack in bitcoin digital currency," *CoRR*, vol. abs/1402.1718, 2014. [Online]. Available: http://arxiv.org/abs/1402.1718

[45] I. Tsabary and I. Eyal, "The gap game," in *Proceedings of the 11th ACM International Systems and Storage Conference*, ser. SYSTOR '18. New York, NY, USA: Association for Computing Machinery, 2018, p. 132. [Online]. Available: https://doi.org/10.1145/3211890.3211905

[46] C. Feng and J. Niu, "Selfish mining in ethereum," in *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, 2019, pp. 1306–1316.

[47] K. Nayak, S. Kumar, A. Miller, and E. Shi, "Stubborn mining: Generalizing selfish mining and combining with an eclipse attack," in *2016 IEEE European Symposium on Security and Privacy (EuroS&P)*, 2016, pp. 305–320.

[48] A. Judmayer, N. Stifter, A. Zamyatin, I. Tsabary, I. Eyal, P. Gaži, S. Meiklejohn, and E. Weippl, "Pay to win: Cheap, cross-chain bribing attacks on pow cryptocurrencies," in *Financial Cryptography and Data Security. FC 2021 International Workshops*, M. Bernhard, A. Bracciali, L. Gudgeon, T. Haines, A. Klages-Mundt, S. Matsuo, D. Perez, M. Sala, and S. Werner, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2021, pp. 533–549.

[49] J. Bonneau, "Why buy when you can rent?" in *Financial Cryptography and Data Security*, J. Clark, S. Meiklejohn, P. Y. Ryan, D. Wallach, M. Brenner, and K. Rohloff, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2016, pp. 19–26.

[50] S. Gao, Z. Li, Z. Peng, and B. Xiao, "Power adjusting and bribery racing: Novel mining attacks in the bitcoin system," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 833–850. [Online]. Available: https://doi.org/10.1145/3319535.3354203

[51] Z. Yang, C. Yin, J. Ke, T. T. A. Dinh, and J. Zhou, "If you can't beat them, pay them: Bitcoin protection racket is profitable," in *Proceedings of the 38th Annual Computer Security Applications Conference*, ser. ACSAC '22. New York, NY, USA: Association for Computing Machinery, 2022, p. 727–741. [Online]. Available: https://doi.org/10.1145/3564625.3567983

[52] D. Karakostas, A. Kiayias, and T. Zacharias, "Blockchain bribing attacks and the efficacy of counterincentives," 2024. [Online]. Available: https://arxiv.org/abs/2402.06352

[53] A. Yaish, M. Dotan, K. Qin, A. Zohar, and A. Gervais, "Suboptimality in defi," Cryptology ePrint Archive, Paper 2023/892, 2023. [Online]. Available: https://eprint.iacr.org/2023/892

[54] L. Zhou, X. Xiong, J. Ernstberger, S. Chaliasos, Z. Wang, Y. Wang, K. Qin, R. Wattenhofer, D. Song, and A. Gervais, "Sok: Decentralized finance (defi) attacks," in *2023 IEEE Symposium on Security and Privacy (SP)*, 2023, pp. 2444–2461.

[55] M. Mirkin, Y. Ji, J. Pang, A. Klages-Mundt, I. Eyal, and A. Juels, "Bdos: Blockchain denial-of-service," in *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 601–619. [Online]. Available: https://doi.org/10.1145/3372297.3417247

[56] Q. Wang, C. Li, T. Xia, Y. Ren, D. Wang, G. Zhang, and K.-K. R. Choo, "Optimal selfish mining-based denial-of-service attack," *IEEE Transactions on Information Forensics and Security*, vol. 19, pp. 835–850, 2024.

[57] Q. Wang, T. Xia, D. Wang, Y. Ren, G. Miao, and K.-K. R. Choo, "Sdos: Selfish mining-based denial-of-service attack," *IEEE Transactions on Information Forensics and Security*, vol. 17, pp. 3335–3349, 2022.

[58] K. Li, Y. Wang, and Y. Tang, "Deter: Denial of ethereum txpool services," in *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '21. New York, NY, USA: Association for Computing Machinery, 2021, p. 1645–1667. [Online]. Available: https://doi.org/10.1145/3460120.3485369

[59] M. Zhang, R. Li, and S. Duan, "Max attestation matters: Making honest parties lose their incentives in ethereum PoS," in *33rd USENIX Security Symposium (USENIX Security 24)*. Philadelphia, PA: USENIX Association, Aug. 2024, pp. 6255–6272. [Online]. Available: https://www.usenix.org/conference/usenixsecurity24/presentation/zhang-mingfei

[60] J. Neu, E. N. Tas, and D. Tse, "Ebb-and-flow protocols: A resolution of the availability-finality dilemma," in *2021 IEEE Symposium on Security and Privacy (SP)*, 2021, pp. 446–465.

[61] E. N. T. Joachim Neu and D. Tse, "Two attacks on proof-of-stake ghost/ethereum," 2022. [Online]. Available: https://arxiv.org/abs/2203.01315

[62] M. Neuder, D. J. Moroz, R. Rao, and D. C. Parkes, "Selfish behavior in the tezos proof-of-stake protocol," *Cryptoeconomic Systems*, vol. 0, no. 1, apr 5 2021, https://cryptoeconomicsystems.pubpub.org/pub/neuder-selfish-behavior-tezos.

[63] J. Brown-Cohen, A. Narayanan, A. Psomas, and S. M. Weinberg, "Formal barriers to longest-chain proof-of-stake protocols," in *Proceedings of the 2019 ACM Conference on Economics and Computation*, ser. EC '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 459–473. [Online]. Available: https://doi.org/10.1145/3328526.3329567

[64] A. Kiayias, A. Russell, B. David, and R. Oliynykov, "Ouroboros: A provably secure proof-of-stake blockchain protocol," in *2019 IEEE Symposium on Security and Privacy*. San Francisco, CA, USA: IEEE, 2017, pp. 157–174, https://doi.org/10.1109/SP.2019.00063.

[65] W. Li, S. Andreina, J.-M. Bohli, and G. Karame, "Securing proof-of-stake blockchain protocols," in *Data Privacy Management, Cryptocurrencies and Blockchain Technology*, J. Garcia-Alfaro, G. Navarro-Arribas, H. Hartenstein, and J. Herrera-Joancomartí, Eds. Cham: Springer International Publishing, 2017, pp. 297–315.

[66] B. Yee, "Keep your transactions on short leashes," 2022. [Online]. Available: https://arxiv.org/abs/2206.11974

[67] S. Azouvi and M. Vukolić, "Pikachu: Securing pos blockchains from long-range attacks by checkpointing into bitcoin pow using taproot," in *Proceedings of the 2022 ACM Workshop on Developments in Consensus*, ser. ConsensusDay '22. New York, NY, USA: Association for Computing Machinery, 2022, p. 53–65. [Online]. Available: https://doi.org/10.1145/3560829.3563563

[68] F. Luo, H. Lin, Z. Li, X. Luo, R. Luo, Z. He, S. Song, T. Chen, and W. Luo, "Towards automatic discovery of denial-of-service weaknesses in blockchain resource models," in *Proceedings of the 31st ACM Conference on Computer and Communications Security (CCS)*, Oct. 2024, aCM Conference on Computer and Communications Security (CCS 2024) ; Conference date: 14-10-2024.

[69] Y. Wang, Y. Tang, K. Li, W. Ding, and Z. Yang, "Understanding ethereum mempool security under asymmetric DoS by symbolized stateful fuzzing," in *33rd USENIX Security Symposium (USENIX Security 24)*. Philadelphia, PA: USENIX Association, Aug. 2024, pp. 4747–4764. [Online]. Available: https://www.usenix.org/conference/usenixsecurity24/presentation/wang-yibo

[70] A. Yaish, K. Qin, L. Zhou, A. Zohar, and A. Gervais, "Speculative Denial-of-Service attacks in ethereum," in *33rd USENIX Security Symposium (USENIX Security 24)*. Philadelphia, PA: USENIX Association, Aug. 2024, pp. 3531–3548. [Online]. Available: https://www.usenix.org/conference/usenixsecurity24/presentation/yaish

[71] C. Schwarz-Schilling, J. Neu, B. Monnot, A. Asgaonkar, E. N. Tas, and D. Tse, "Three attacks on proof-of-stake ethereum," in *Financial Cryptography and Data Security*, I. Eyal and J. Garay, Eds. Cham: Springer International Publishing, 2022, pp. 560–576.

[72] N. Ruaro, F. Gritti, R. McLaughlin, I. Grishchenko, C. Kruegel, and G. Vigna, "Not your type! detecting storage collision vulnerabilities in ethereum smart contracts," in *31st Annual Network and Distributed System Security Symposium, NDSS 2024, San Diego, California, USA, February 26 - March 1, 2024*. The Internet Society, 2024.

[73] W. Zhang, Z. Zhang, Q. Shi, L. Liu, L. Wei, Y. Liu, X. Zhang, and S. Cheung, "Nyx: Detecting exploitable front-running vulnerabilities in smart contracts," in *IEEE Symposium on Security and Privacy, SP 2024, San Francisco, CA, USA, May 19-23, 2024*. IEEE, 2024, pp. 2198–2216. [Online]. Available: https://doi.org/10.1109/SP54263.2024.00146

[74] C. Sendner, L. Petzi, J. Stang, and A. Dmitrienko, "Large-scale study of vulnerability scanners for ethereum smart contracts," in *2024 IEEE Symposium on Security and Privacy (SP)*. Los Alamitos, CA, USA: IEEE Computer Society, May 2024, pp. 2273–2290. [Online]. Available: https://doi.ieeecomputersociety.org/10.1109/SP54263.2024.00230

[75] Z. He, Z. Li, A. Qiao, X. Luo, X. Zhang, T. Chen, S. Song, D. Liu, and W. Niu, "Nurgle: Exacerbating resource consumption in blockchain state storage via mpt manipulation," in *2024 IEEE Symposium on Security and Privacy (SP)*, 2024, pp. 2180–2197.

Appendix A. Related Work

We now embark on a discussion concerning the related work of this paper, primarily focusing on three aspects:

attacks targeting vulnerabilities in incentive models and consensus protocols of Nakamoto-like blockchains, Ethereum 2.0-like blockchains, and other facets.

A.1. Attacks on Incentive Model and Consensus Protocol of Nakamoto-like Blockchains

Withholding Attack. In Nakamoto-like blockchains, Withholding Attack represents a prevalent malicious behavior [34], [40], [41], [42], [43], [44], [45], [46]. Most consensus-layer attacks rely on block withholding and specific capabilities of the attacker, exemplified by the Selfish Mining attack, where the adversary withholds blocks and strategically chooses the timing of their release to increase their share of blocks. Variants such as Stubborn Mining attacks involve the adversary continuing to mine even when their selfish fork lags behind the honest chain [41], [47]. These attacks undermine the fairness and normal operation of blockchain networks. By impeding the timely dissemination of certain blocks, adversaries attempt to control the pace of block generation within the network and disrupt the normal functioning of honest nodes. This not only results in the potential waste of computing power and resources of honest nodes but also complicates and destabilizes the network's fork situation, reducing its security and reliability. Consequently, user trust in Nakamoto-like blockchains is eroded, exerting a negative impact on the entire blockchain ecosystem.

Bribing Attack. In Nakamoto-like blockchains, the Bribing Attack poses a severe threat to network security and stability [48], [49], [50], [51], [52]. First described by Bonneau, this type of attack encompasses multiple mechanisms. Early research indicated that bribery attacks may intensify when participants rely solely on transaction fees for compensation. For instance, bribery attacks implemented through smart contracts exhibit "inbound" and "outbound" modes, with subsequent research continuously enhancing their efficiency, such as through cross-system double-spending conspiracies or leveraging smart contract payments. The motives for these attacks are diverse, including double-spending, consensus disruption, censorship, transaction front-running, or the sabotage of hash-time locked contracts (HTLCs) [53], [54]. Recent research has proposed bribery attack models targeting longest-chain blockchains (such as Bitcoin), with some assuming a dichotomy between honest and rationally bribe-accepting participants, employing "inbound" bribery (paid in native tokens). Other models may be more general, considering rationality on all sides, with "outbound" bribery (e.g., paid in USD) unaffected by token price fluctuations. These attacks significantly disrupt the normal operational order of blockchains.

DoS attack. In Nakamoto-like blockchains, Denial of Service (DoS) attacks primarily disrupt system availability by consuming network resources [55], [56], [57], [58]. At the network layer, adversaries often launch traffic flooding attacks, such as sending a massive number of false requests or packets to target nodes, rapidly consuming network bandwidth and causing nodes to become overwhelmed with processing these invalid flows, thereby preventing them from timely responding to legitimate transactions. For example, a deluge of spam transaction requests can clog network channels, hindering the smooth propagation and processing of normal transactions. Such attacks significantly interfere with the normal operation of blockchain networks, increasing transaction confirmation times and reducing system efficiency. They undermine the decentralized feature of blockchain, as nodes under attack are unable to participate equally in network activities. Furthermore, DoS attacks may trigger a chain reaction, affecting users' trust in blockchain and impeding its application and development across various domains.

Timestamp Attack. In Nakamoto-like blockchains, Timestamp Manipulation Attacks exploit the configurable feature of block timestamps, allowing adversaries to meticulously set timestamps that may interfere with the normal generation and confirmation processes of blocks [16]. For instance, adversaries can manipulate mining difficulty adjustments by setting unreasonable timestamps, thereby impacting their mining rewards or network stability. This attack vector potentially undermines the fairness and normal operational order of blockchain, granting adversaries undue advantages and posing a latent threat to the security and reliability of Nakamoto-like blockchains. Some research has addressed the role of timestamps in attacks, such as in Ethereum where timestamps were used to attack smart contracts. However, there is limited research on timestamp attacks at the consensus layer. Stretch identified two timestamp vulnerabilities in the geth codebase, one of which could be used to reduce honest mining difficulty, but did not delve into the impact on consensus mechanisms.

It is worth mention that the UUM attack proposed in our paper falls within the category of timestamp attacks, while SUUM combines withholding attacks with timestamp attacks.

A.2. Attacks on Incentive Model and Consensus Protocol of Ethereum 2.0-like Blockchains

Withholding Attack. In Ethereum 2.0-like PoS blockchains, the Withholding Attack poses a threat to network stability. An adversary may withhold blocks to prevent honest nodes from propagating them, releasing the withheld blocks at an opportune moment to gain undue benefits [59], [60], [61], [62]. This attack resembles the withholding attack observed in Nakamoto-style blockchains but exhibits distinct characteristics due to the unique protocol mechanisms of Ethereum 2.0 PoS. For instance, adversaries can exploit the staking and voting mechanisms of validator nodes to disrupt the normal block confirmation process during consensus. By withholding blocks they have mined or validated, they can influence the block height and fork conditions within the network, thereby undermining network consistency and certainty. Such attacks may result in wasted efforts by honest nodes, increase network uncertainty, and diminish user trust in the network. Furthermore, they may afford the adversary

an advantageous position in subsequent block competitions or network decisions, thereby impacting the fairness and security of the entire Ethereum 2.0-like blockchain network.

Nothing-at-stake Attack. In Ethereum 2.0-like PoS blockchains, the Nothing-at-Stake Attack represents a severe threat to network activity and performance [63], [64], [65]. Due to the characteristics of the staking distribution and validation mechanisms within the PoS system, adversaries can exploit this vulnerability to contribute effortlessly to multiple forks. Without bearing any actual risk, adversaries can interfere with the normal transaction confirmation and block production processes by operating on different forks. This leads to a chaotic situation of network forks, making it difficult for the system to achieve a unified consensus and reducing the network's efficiency in processing transactions. For example, under normal circumstances, validators should choose to support a single correct fork based on their own staking and network rules. However, adversaries can freely stake on multiple forks, disrupting the stability and reliability of the network. This, in turn, affects users' trust in the Ethereum 2.0-like blockchain network, hinders its normal development, and has a negative impact on the entire blockchain ecosystem.

Long-range Attack. The Long-Range Attack is a potential threat to Ethereum 2.0-like PoS blockchains, exploiting a vulnerability in the PoS mechanism where adversaries can secretly construct an alternative chain starting from an earlier point in the blockchain history [64], [66], [67]. This attack is named for the adversaries' ability to backtrack to a "long-range" past point in the blockchain without significant resource consumption and initiate a new chain from there. If successful, the previous transaction records and states of the network will become untrusted, undermining the consistency principle of blockchain data. For instance, adversaries can alter transaction information in past blocks, reverse completed transaction states, or create new transactions out of thin air, severely disrupting the normal operational order of the network. This attack leverages potential flaws in Ethereum 2.0-like PoS mechanism in managing historical data, dealing a severe blow to the trust foundation of the entire blockchain ecosystem. It may spark user concerns about network security, thereby impacting the widespread adoption and sustainable development of Ethereum 2.0-like PoS blockchains. Consequently, continuous enhancements in preventive measures are necessary to defend against such attacks.

DoS Attack. In Ethereum 2.0-like PoS blockchains, DoS attacks manifest in various forms and pose severe threats. At the network layer, DoS attacks resemble traditional patterns, where adversaries launch flood attacks by sending a vast number of fake requests or data packets to occupy network bandwidth and node resources, thereby preventing legitimate transactions from being processed normally. Attacks at the application layer primarily target smart contracts and transaction pools (Txpool) [68], [69], [70]. At the smart contract level, adversaries exploit vulnerabilities or design flaws in contracts to initiate malicious transactions that cause contract execution to enter infinite loops or exhaust computational resources. For example, triggering complex logic operations without resource limits can lead to contract execution timeouts and node resource depletion, resulting in a DoS condition that affects the normal operation of the network. In terms of the Txpool, adversaries send a large number of transactions with low gas prices but high computational complexity, causing congestion in the Txpool and preventing normal transactions from being processed in a timely manner. This can also manipulate the order of transactions, influence participants' choices in packaging transactions, interfere with the normal entry of transactions into the blockchain network, and undermine the fairness and efficiency of transaction processing in the network.

Reorg Attack. The Reorg Attack in Ethereum 2.0-like PoS blockchains poses a severe threat to network security and stability [59], [71]. This attack increases the proportion of blocks belonging to Byzantine validators on the main chain to gain more profits or downgrade chain quality or performance. It encompasses Short Reorg Attacks (SRA) and Long Reorg Attacks (LRA). By leveraging controlled validator nodes, adversaries release new block branches at specific times, causing the original main chain to be replaced. This results in the transaction states on the previous main chain becoming uncertain, affecting user fund security and transaction reliability, disrupting block order and transaction confirmation processes, and weakening users' trust in the network.

Grinding Attack. The Grinding Attack in Ethereum 2.0-like PoS blockchains is a malicious act that exploits vulnerabilities in the system's mechanisms [63]. Adversaries primarily manipulate the random number generation to influence the selection of block proposers, leveraging the characteristics of the Ethereum Virtual Machine (EVM), such as recursive calls and storage operations, to meticulously construct malicious contracts. These malicious contracts are designed to exhaust the memory, storage, or computational resources of nodes, thereby disrupting the fairness and normal operation of the system. For instance, adversaries may continuously attempt to generate random numbers favorable to themselves, increasing their chances of becoming block proposers and subsequently controlling the direction of network development. Alternatively, they can exploit recursive call vulnerabilities in contracts to cause the contracts to fall into an infinite loop of execution, consuming significant node resources and preventing other legitimate transactions and operations from proceeding smoothly. This severely impacts the stability and efficiency of the Ethereum 2.0-like PoS blockchain network, exerting a negative influence on the entire blockchain ecosystem.

A.3. Attacks on Other Aspects

Smart Contract Attack. In Ethereum 2.0-like PoS blockchains, the attack vectors targeting smart contracts are diverse and pose significant risks [72], [73], [74]. Reentrancy vulnerabilities represent one of the common attack methods, where adversaries exploit flaws in the contract's failure to correctly update its state during external calls,

resulting in repeated invocations of contract functions and the abnormal transfer of contract assets. The infamous DAO incident suffered substantial losses due to a reentrancy vulnerability. Overflow vulnerabilities are also noteworthy, as errors in computation may arise when data exceeds the range of variable types, disrupting the normal logical execution of contracts. Furthermore, adversaries exploit vulnerabilities in contract permission management to obtain unauthorized operational privileges, allowing them to tamper with contract data or execute malicious operations. These attacks not only render the contracts ineffective in fulfilling their intended functions but may also lead to user fund losses, transaction data chaos, and severely impact the stability and trust of the Ethereum ecosystem. While formal verification and fuzz testing techniques offer certain defenses to a degree, new attack vectors continue to emerge, necessitating ongoing research and the enhancement of defensive measures.

Txpool Attack. In Ethereum 2.0-like PoS blockchains, attacks against the Txpool primarily exploit its mechanism characteristics to disrupt transaction processing [58], [69]. Adversaries often employ a strategy of submitting a large number of transactions with low gas prices but high computational complexity, thereby causing congestion within the Txpool. Since the Txpool typically sorts and processes transactions based on factors such as gas price, these malicious transactions, despite paying less gas, consume substantial resources for verification and processing due to their computational complexity. This leads to a situation where legitimate transactions cannot be timely packaged by participants, thereby extending transaction confirmation times. Furthermore, adversaries may attempt to manipulate the order of transactions within the Txpool by controlling the timing of transaction broadcasts and their queuing positions, thereby influencing participants' selection of transactions. This enables certain specific transactions to be prioritized or delayed in processing, disrupting the fairness and normal flow of transaction processing. Consequently, this affects the transaction efficiency and user experience of the entire Ethereum 2,0-like blockchain network, posing a threat to the network's normal operation.

Data Storage Attack. In Ethereum 2.0-like PoS blockchains, attacks targeting data storage pose significant threats [75]. Adversaries may exploit vulnerabilities in the storage mechanism to illegally manipulate storage pointers, maliciously tampering with account balances, data, or access control information, resulting in erroneous or inconsistent account states. Additionally, adversaries can leverage the characteristics of the data structures used in account storage by carefully crafting accounts that increase the cost of account access upon insertion, thereby achieving the purpose of raising storage and access cost. For instance, modifying account balance data can lead to abnormal fund displays or illegal transfers; disrupting access controls can grant unauthorized account operation privileges. Such attacks may also interfere with the account state update process, causing account states to become disconnected from actual transaction conditions. If successful, these attacks can lead to severe consequences such as user asset losses and transaction record chaos, undermining the accurate processing and trust mechanisms of account information within the entire blockchain system.

Appendix B. Proof of Theorem 1

The initiation condition for the UUM attack is $\mathcal{D}_1^{p_h} \leq \mathcal{D}_0^{p_h}$. Based on the difficulty calculation formula, we have:

$$\begin{aligned} \mathcal{D}_0^{p_h} + max\left\{1 - \left\lfloor \frac{t_1^{p_h} - t_0^{p_h}}{9} \right\rfloor, -99\right\} \cdot \left\lfloor \frac{\mathcal{D}_0^{p_h}}{2048} \right\rfloor &\leq \mathcal{D}_0^{p_h} \\ max\left\{1 - \left\lfloor \frac{t_1^{p_h} - t_0^{p_h}}{9} \right\rfloor, -99\right\} \cdot \left\lfloor \frac{\mathcal{D}_0^{p_h}}{2048} \right\rfloor &\leq 0 \\ \left\lfloor \frac{t_1^{p_h} - t_0^{p_h}}{9} \right\rfloor &\geq 1. \end{aligned} \tag{3}$$

Hence, we have $\left\lfloor \frac{t_1^{p_h} - t_0^{p_h}}{9} \right\rfloor \in [1, +\infty)$.

Appendix C. Proof of Theorem 2

Firstly, for the adversary's block $\mathcal{B}_i^{p_a}$ to be valid, it is necessary for block $\mathcal{B}_1^{p_a}$ to be generated after its parent block $\mathcal{B}_0^{p_a}$. Consequently, the timestamp $t_1^{p_a}$ of block $\mathcal{B}_1^{p_a}$ should be greater than the timestamp $t_0^{p_a}$ of its parent block $\mathcal{B}_0^{p_a}$, i.e.:

$$t_1^{p_a} - t_0^{p_a} \geq 1. \tag{4}$$

Secondly, only when the difficulty $\mathcal{D}_1^{p_a}$ of the adversary's block $\mathcal{B}_1^{p_a}$ is greater than the difficulty $\mathcal{D}_1^{p_h}$ of the honest block $\mathcal{B}_0^{p_h}$ will other honest participants choose the adversary's block. At this point, we have: $\mathcal{D}_1^{p_a} > \mathcal{D}_1^{p_h}$. Based on the difficulty calculation formula, we can derive the following inequality:

$$\begin{aligned} &\mathcal{D}_0^{p_a} + max\left\{1 - \left\lfloor \frac{t_1^{p_a} - t_0^{p_a}}{9} \right\rfloor, -99\right\} \cdot \left\lfloor \frac{\mathcal{D}_0^{p_a}}{2048} \right\rfloor \\ &> \mathcal{D}_0^{p_h} + max\left\{1 - \left\lfloor \frac{t_1^{p_h} - t_0^{p_h}}{9} \right\rfloor, -99\right\} \cdot \left\lfloor \frac{\mathcal{D}_0^{p_h}}{2048} \right\rfloor. \end{aligned} \tag{5}$$

Since $\mathcal{D}_0^{p_h} = \mathcal{D}_0^{p_a}$ and $t_0^{p_h} = t_0^{p_a}$, we therefore have:

$$\begin{aligned} &max\left\{1 - \left\lfloor \frac{t_1^{p_a} - t_0^{p_a}}{9} \right\rfloor, -99\right\} \\ &> max\left\{1 - \left\lfloor \frac{t_1^{p_h} - t_0^{p_a}}{9} \right\rfloor, -99\right\}. \end{aligned} \tag{6}$$

The above inequality can be transformed into the following system of inequalities:

$$\begin{cases} 1 - \left\lfloor \frac{t_1^{p_a} - t_0^{p_a}}{9} \right\rfloor > 1 - \left\lfloor \frac{t_1^{p_h} - t_0^{p_a}}{9} \right\rfloor & (1) \\ 1 - \left\lfloor \frac{t_1^{p_a} - t_0^{p_a}}{9} \right\rfloor > -99 & (2) \end{cases}. \tag{7}$$

Based on Inequality (1), we have:

$$\left\lfloor \frac{t_1^{p_h} - t_0^{p_a}}{9} \right\rfloor - \left\lfloor \frac{t_1^{p_a} - t_0^{p_a}}{9} \right\rfloor > 0. \tag{8}$$

Further rearranging the above inequality, we can derive:

$$\left\lfloor \frac{t_1^{p_h} - t_1^{p_a}}{9} \right\rfloor > 0. \tag{9}$$

Since $\left\lfloor \frac{t_1^{p_h} - t_1^{p_a}}{9} \right\rfloor$ can only take integer values, i.e., $\left\lfloor \frac{t_1^{p_h} - t_1^{p_a}}{9} \right\rfloor \in N$, we therefore have:

$$\left\lfloor \frac{t_1^{p_h} - t_1^{p_a}}{9} \right\rfloor \geq 1. \tag{10}$$

Based on the inequality above, we can get $\left\lfloor \frac{t_1^{p_h} - t_1^{p_a}}{9} \right\rfloor \in [1, +\infty)$.

Based on the Inequality (10), we can derive:

$$t_1^{p_h} - t_1^{p_a} \geq 9. \tag{11}$$

Based on the Inequality (2), we can derive:

$$\left\lfloor \frac{t_1^{p_a} - t_0^{p_a}}{9} \right\rfloor < 100 \tag{12}$$

Since $\left\lfloor \frac{t_1^{p_a} - t_0^{p_a}}{9} \right\rfloor$ can only take integer values, i.e., $\left\lfloor \frac{t_1^{p_a} - t_0^{p_a}}{9} \right\rfloor \in N$, we therefore have: $\left\lfloor \frac{t_1^{p_a} - t_0^{p_a}}{9} \right\rfloor \leq 99$. Based on the aforementioned inequalities, we have:

$$t_1^{p_a} - t_0^{p_a} < 900. \tag{13}$$

Combining Inequality (4) and Inequality (13), we obtain: $1 \leq t_1^{p_a} - t_0^{p_a} < 900$, which implies that $t_1^{p_a} - t_0^{p_a} \in [1, 900)$.

Appendix D. Proof of Theorem 3

According to Theorem 2, the conditions for a successful UUM attack are: $\left\lfloor \frac{t_1^{p_h} - t_1^{p_a}}{9} \right\rfloor \in [1, +\infty)$ and $t_1^{p_a} - t_0^{p_a} \in [1, 900)$. To minimize the risk associated with the UUM attack, which corresponds to minimizing the block difficulty growth rate, Theorem 2 suggests that we take the minimum value within the condition $\left\lfloor \frac{t_1^{p_h} - t_1^{p_a}}{9} \right\rfloor \in [1, +\infty)$, i.e., $\left\lfloor \frac{t_1^{p_h} - t_1^{p_a}}{9} \right\rfloor = 1$. This implies that $t_1^{p_h} - t_1^{p_a} \in [9, 18)$. By integrating these two conditions, we can prove the theorem.

Appendix E. Proof of Theorem 4

The condition for the RUM attack is that mining on top of the former block incurs no additional risk, i.e., $\mathcal{D}_1^{p_h} = \mathcal{D}_0^{p_h}$. Based on the difficulty calculation formula, we have:

$$\begin{aligned} \mathcal{D}_0^{p_h} + max\left\{1 - \left\lfloor \frac{t_1^{p_h} - t_0^{p_h}}{9} \right\rfloor, -99\right\} \cdot \left\lfloor \frac{\mathcal{D}_0^{p_h}}{2048} \right\rfloor &= \mathcal{D}_0^{p_h} \\ max\left\{1 - \left\lfloor \frac{t_1^{p_h} - t_0^{p_h}}{9} \right\rfloor, -99\right\} &= 0 \\ \left\lfloor \frac{t_1^{p_h} - t_0^{p_h}}{9} \right\rfloor &= 1. \end{aligned} \tag{14}$$

TABLE 1. UUM STATE TRANSITION.

State	Destination	Transition	Probability	Reward	
				$\mathcal{P}_\mathcal{H}$	$\mathcal{P}_\mathcal{S}$
Deployment	Deployment	$\mathcal{P}_\mathcal{A}$ $any\ t$	$\mathbb{P}_\mathcal{A}(any\ t)$	0	1
Deployment	Deployment	$\mathcal{P}_\mathcal{H}$ $t < 9$	$\mathbb{P}_\mathcal{H}(t < 9)$	1	0
Deployment	Attack	$\mathcal{P}_\mathcal{H}$ $t \geq 9$	$\mathbb{P}_\mathcal{H}(t \geq 9)$	1	0
Attack	Attack	$\mathcal{P}_\mathcal{H}$ $t \geq 9$	$\mathbb{P}_\mathcal{H}(t \geq 9)$	1	0
Attack	Deployment	$\mathcal{P}_\mathcal{H}$ $t < 9$	$\mathbb{P}_\mathcal{H}(t < 9)$	1	0
Attack	Deployment	$\mathcal{P}_\mathcal{A}$ $any\ t$	$\mathbb{P}_\mathcal{A}(any\ t)$	-1	1

Appendix F. Proof of Theorem 5

Based on the analysis of state transitions for UUM deployment and attack states presented in Section 4, we calculate the absolute and relative shares of coin-base rewards for both UUM adversaries and honest participants.

The absolute share of coin-base rewards for UUM adversary $\mathcal{P}_\mathcal{A}$ is:

$$\begin{aligned} \mathbb{R}_{\mathcal{P}_\mathcal{A}}^{UUM} &= \mathbb{P}\left(Deploy\right) \cdot \mathbb{P}_\mathcal{A}\left(any\ t\right) + \mathbb{P}\left(Attack\right) \cdot \mathbb{P}_\mathcal{A}\left(any\ t\right) \\ &= \left(\mathbb{P}\left(Deploy\right) + \mathbb{P}\left(Attack\right)\right) \cdot \mathbb{P}_\mathcal{A}\left(any\ t\right) \\ &= \mathbb{P}_\mathcal{A}\left(any\ t\right). \end{aligned} \tag{15}$$

Therefore, the absolute share of coin-base rewards for UUM adversary $\mathcal{P}_\mathcal{A}$ remains unchanged.

The absolute share of coin-base rewards for honest participant $\mathcal{P}_\mathcal{H}$ is:

$$\begin{aligned} \mathbb{R}_{\mathcal{P}_\mathcal{H}}^{UUM} &= \mathbb{P}\left(Deploy\right) \cdot \mathcal{P}_\mathcal{H}\left(t < 9\right) + \mathbb{P}\left(Deploy\right) \cdot \mathbb{P}_\mathcal{H}\left(t \geq 9\right) \\ &+ \mathbb{P}\left(Attack\right) \cdot \mathbb{P}_\mathcal{H}\left(t \geq 9\right) + \mathbb{P}\left(Attack\right) \cdot \mathbb{P}_\mathcal{H}\left(t < 9\right) \\ &- \mathbb{P}\left(Attack\right) \cdot \mathbb{P}_\mathcal{A}\left(any\ t\right) \\ &= \mathbb{P}\left(Deploy\right) \cdot \left(\mathcal{P}_\mathcal{H}\left(t < 9\right) + \mathbb{P}_\mathcal{H}\left(t \geq 9\right)\right) \\ &+ \mathbb{P}\left(Attack\right) \cdot \left(\mathbb{P}_\mathcal{H}\left(t \geq 9\right) + \mathbb{P}_\mathcal{H}\left(t < 9\right)\right) \\ &- \mathbb{P}\left(Attack\right) \cdot \mathbb{P}_\mathcal{A}\left(any\ t\right) \\ &= \mathbb{P}\left(Deploy\right) \cdot \mathcal{P}_\mathcal{H}\left(any\ t\right) + \mathbb{P}\left(Attack\right) \cdot \mathcal{P}_\mathcal{H}\left(any\ t\right) \\ &- \mathbb{P}\left(Attack\right) \cdot \mathbb{P}_\mathcal{A}\left(any\ t\right) \\ &= \left(\mathbb{P}\left(Deploy\right) + \mathbb{P}\left(Attack\right)\right) \cdot \mathcal{P}_\mathcal{H}\left(any\ t\right) \\ &- \mathbb{P}\left(Attack\right) \cdot \mathbb{P}_\mathcal{A}\left(any\ t\right) \\ &= \mathcal{P}_\mathcal{H}\left(any\ t\right) - \mathbb{P}\left(Attack\right) \cdot \mathbb{P}_\mathcal{A}\left(any\ t\right) \\ &< \mathcal{P}_\mathcal{H}\left(any\ t\right). \end{aligned} \tag{16}$$

Therefore, the absolute share of coin-base rewards for honest participant $\mathcal{P}_\mathcal{H}$ decreases.

The expected relative share of coin-base rewards for UUM adversary $\mathcal{P}_\mathcal{A}$ is:

$$\begin{aligned}\mathbb{E}\left[\mathbb{R}_{\mathcal{P}_\mathcal{A}}^{UUM}\right] &= \frac{\mathbb{R}_{\mathcal{P}_\mathcal{A}}^{UUM}}{\mathbb{R}_{\mathcal{P}_\mathcal{A}}^{UUM}+\mathbb{R}_{\mathcal{P}_\mathcal{H}}^{UUM}} \\ &= \frac{\mathbb{P}_\mathcal{A}\left(any\ t\right)}{\begin{pmatrix}\mathbb{P}_\mathcal{A}\left(any\ t\right)+\mathcal{P}_\mathcal{H}\left(any\ t\right)\\ -P\left(Attack\right)\cdot\mathbb{P}_\mathcal{A}\left(any\ t\right)\end{pmatrix}} \\ &> \frac{\mathbb{P}_\mathcal{A}\left(any\ t\right)}{\mathbb{P}_\mathcal{A}\left(any\ t\right)+\mathcal{P}_\mathcal{H}\left(any\ t\right)} \\ &= \mathbb{P}_\mathcal{A}\left(any\ t\right).\end{aligned} \tag{17}$$

Therefore, the expected relative share of coin-base rewards for UUM adversary $\mathcal{P}_\mathcal{A}$ increases.

The expected relative share of coin-base rewards for honest participant $\mathcal{P}_\mathcal{H}$ is:

$$\begin{aligned}\mathbb{E}\left[\mathbb{R}_{\mathcal{P}_\mathcal{H}}^{UUM}\right] &= \frac{\mathbb{R}_{\mathcal{P}_\mathcal{H}}^{UUM}}{\mathbb{R}_{\mathcal{P}_\mathcal{A}}^{UUM}+\mathbb{R}_{\mathcal{P}_\mathcal{H}}^{UUM}} \\ &= \frac{\mathcal{P}_\mathcal{H}\left(any\ t\right)-\mathbb{P}\left(Attack\right)\cdot\mathbb{P}_\mathcal{A}\left(any\ t\right)}{\begin{pmatrix}\mathbb{P}_\mathcal{A}\left(any\ t\right)+\mathcal{P}_\mathcal{H}\left(any\ t\right)\\ -\mathbb{P}\left(Attack\right)\cdot\mathbb{P}_\mathcal{A}\left(any\ t\right)\end{pmatrix}} \\ &< \frac{\mathcal{P}_\mathcal{H}\left(any\ t\right)}{\mathbb{P}_\mathcal{A}\left(any\ t\right)+\mathcal{P}_\mathcal{H}\left(any\ t\right)} \\ &= \mathcal{P}_\mathcal{H}\left(any\ t\right).\end{aligned} \tag{18}$$

Therefore, the expected relative share of coin-base rewards for honest participant $\mathcal{P}_\mathcal{H}$ decreases.

Appendix G. Proof of Theorem 6

The timestamps of the honest block $\mathcal{B}_i^{p_h}$ and the SUUM adversary's block $\mathcal{B}_i^{p_a}$ at the same height i are denoted as $t_i^{p_h}$ and $t_i^{p_a}$, respectively, with corresponding difficulties $\mathcal{D}_i^{p_h}$ and $\mathcal{D}_i^{p_a}$. For the RUM attack to be successful, the difficulty $\mathcal{D}_i^{p_a}$ of the SUUM adversary's block $\mathcal{B}_i^{p_a}$ at the same height should be greater than the difficulty $\mathcal{D}_i^{p_h}$ of the honest block $\mathcal{B}_i^{p_h}$. According to the difficulty calculation formula, we have:

$$\mathcal{D}_i^{p_h} = \mathcal{D}_{i-1}^{p_h} + max\left\{1-\left[\frac{t_i^{p_h}-t_{i-1}^{p_h}}{9}\right],-99\right\}\cdot\left\lfloor\frac{\mathcal{D}_{i-1}^{p_h}}{2048}\right\rfloor \tag{19}$$

and

$$\mathcal{D}_i^{p_a} = \mathcal{D}_{i-1}^{p_a} + max\left\{1-\left[\frac{t_i^{p_a}-t_{i-1}^{p_a}}{9}\right],-99\right\}\cdot\left\lfloor\frac{\mathcal{D}_{i-1}^{p_a}}{2048}\right\rfloor. \tag{20}$$

Given that $t_0^{p_h}=t_0^{p_a}$ and $\mathcal{D}_0^{p_h}=\mathcal{D}_0^{p_a}$ (representing the same block), for $i=1,\ 2\ ,\ldots,\ n$, it is necessary to ensure that $\mathcal{D}_i^{p_a}>\mathcal{D}_i^{p_h}$ holds true consistently.

(1) For $i=1$, we need to ensure that the following condition holds:

$$\mathcal{D}_i^{p_a}>\mathcal{D}_i^{p_h}\Rightarrow\mathcal{D}_1^{p_a}>\mathcal{D}_1^{p_h}. \tag{21}$$

According to the difficulty calculation formula, we have:

$$\begin{aligned}&\mathcal{D}_0^{p_a}+max\left\{1-\left[\frac{t_1^{p_a}-t_0^{p_a}}{9}\right],-99\right\}\cdot\left\lfloor\frac{\mathcal{D}_0^{p_a}}{2048}\right\rfloor \\ &>\mathcal{D}_0^{p_h}+max\left\{1-\left[\frac{t_1^{p_h}-t_0^{p_h}}{9}\right],-99\right\}\cdot\left\lfloor\frac{\mathcal{D}_0^{p_h}}{2048}\right\rfloor \\ &\Rightarrow max\left\{1-\left[\frac{t_1^{p_a}-t_0^{p_a}}{9}\right],-99\right\} \\ &>max\left\{1-\left[\frac{t_1^{p_h}-t_0^{p_h}}{9}\right],-99\right\}.\end{aligned} \tag{22}$$

By rearranging this inequality, we obtain the following system of inequalities:

$$\begin{cases}1-\left\lfloor\frac{t_1^{p_a}-t_0^{p_a}}{9}\right\rfloor>1-\left\lfloor\frac{t_1^{p_h}-t_0^{p_a}}{9}\right\rfloor & (1)\\ 1-\left\lfloor\frac{t_1^{p_a}-t_0^{p_a}}{9}\right\rfloor>-99 & (2)\end{cases}. \tag{23}$$

According to inequality (1), we have:

$$\begin{aligned}t_1^{p_h}-t_0^{p_h}-(t_1^{p_a}-t_0^{p_a})\geq 9 \\ \Rightarrow t_1^{p_h}-t_1^{p_a}\geq 9\end{aligned}. \tag{24}$$

Thus, we have:

$$\left\lfloor\frac{t_1^{p_h}-t_1^{p_a}}{9}\right\rfloor\in[1,+\infty). \tag{25}$$

According to inequality (2), we have:

$$\left\lfloor\frac{t_1^{p_a}-t_0^{p_a}}{9}\right\rfloor<100. \tag{26}$$

According to this inequality, we obtain $t_1^{p_a}-t_0^{p_a}<900$. In order for a block $\mathcal{B}_1^{p_a}$ to be valid, we derive $t_1^{p_a}-t_0^{p_a}>0$. Since $t_1^{p_a}-t_0^{p_a}\in N$, we have:

$$t_1^{p_a}-t_0^{p_a}\in(1,900). \tag{27}$$

According to equations (25) and (27), Theorem 6 (1) is proved. Below we prove Theorem 6 (2).

(2) Similarly, for $i=2$, we need to ensure that the following condition holds:

$$\mathcal{D}_i^{p_a}>\mathcal{D}_i^{p_h}\Rightarrow\mathcal{D}_2^{p_a}>\mathcal{D}_2^{p_h}. \tag{28}$$

According to the difficulty calculation formula, we have:

$$\mathcal{D}_2^{p_h} = \mathcal{D}_1^{p_h} + max\left\{1-\left\lfloor\frac{t_2^{p_h}-t_1^{p_h}}{9}\right\rfloor,-99\right\}\cdot\left\lfloor\frac{\mathcal{D}_1^{p_h}}{2048}\right\rfloor \tag{29}$$

and

$$\mathcal{D}_2^{p_a} = \mathcal{D}_1^{p_a} + max\left\{1-\left\lfloor\frac{t_2^{p_a}-t_1^{p_a}}{9}\right\rfloor,-99\right\}\cdot\left\lfloor\frac{\mathcal{D}_1^{p_a}}{2048}\right\rfloor. \tag{30}$$

Therefore, we can derive:

$$\begin{aligned}\mathcal{D}_2^{p_a} - \mathcal{D}_2^{p_h} &= \overbrace{(\mathcal{D}_1^{p_a} - \mathcal{D}_1^{p_h})}^{>0} \\ &+ max\left\{1 - \left\lfloor \frac{t_2^{p_a} - t_1^{p_a}}{9} \right\rfloor, -99\right\} \\ &- max\left\{1 - \left\lfloor \frac{t_2^{p_h} - t_1^{p_h}}{9} \right\rfloor, -99\right\} \geq 0.\end{aligned} \tag{31}$$

We take the example of a minimum-risk attack (where the difficulty of an adversarial block at the same height in the SUUM model is one greater than that of an honest block).

(2.1) For $\left\lfloor \frac{\mathcal{D}_1^{p_a}}{2048} \right\rfloor = \left\lfloor \frac{\mathcal{D}_1^{p_h}}{2048} \right\rfloor$, it suffices to ensure that the following condition holds:

$$\begin{aligned}&max\left\{1 - \left\lfloor \frac{t_2^{p_a} - t_1^{p_a}}{9} \right\rfloor, -99\right\} \\ &- max\left\{1 - \left\lfloor \frac{t_2^{p_h} - t_1^{p_h}}{9} \right\rfloor, -99\right\} \geq 0.\end{aligned} \tag{32}$$

Convert the above inequality into the following system of inequalities:

$$\begin{cases} 1 - \left\lfloor \frac{t_2^{p_a} - t_1^{p_a}}{9} \right\rfloor \geq 1 - \left\lfloor \frac{t_2^{p_h} - t_1^{p_h}}{9} \right\rfloor & (1) \\ 1 - \left\lfloor \frac{t_2^{p_a} - t_1^{p_a}}{9} \right\rfloor \geq -99 & (2) \end{cases}. \tag{33}$$

To solve inequality (1), we have:

$$\left\lfloor \frac{t_2^{p_h} - t_1^{p_h} - (t_2^{p_a} - t_1^{p_a})}{9} \right\rfloor \geq 0. \tag{34}$$

To solve inequality (2), we have:

$$\left\lfloor \frac{t_2^{p_a} - t_1^{p_a}}{9} \right\rfloor \leq 100. \tag{35}$$

Therefore, the solution to the above system of inequalities is $\left\lfloor \frac{t_2^{p_h} - t_1^{p_h} - \left(t_2^{p_a} - t_1^{p_a}\right)}{9} \right\rfloor \geq 0$ and $\left\lfloor \frac{t_2^{p_a} - t_1^{p_a}}{9} \right\rfloor \leq 100$. To ensure the legitimacy of block $\mathcal{B}_2^{p_a}$, the following conditions must be satisfied: $t_2^{p_a} - t_1^{p_a} > 0$. Therefore, if $i = 2$ and $\left\lfloor \frac{\mathcal{D}_1^{p_a}}{2048} \right\rfloor = \left\lfloor \frac{\mathcal{D}_1^{p_h}}{2048} \right\rfloor$, the SUUM attack is successful if and only if the following conditions hold: $\left\lfloor \frac{t_2^{p_h} - t_1^{p_h} - \left(t_2^{p_a} - t_1^{p_a}\right)}{9} \right\rfloor \geq 0$ and $t_2^{p_a} - t_1^{p_a} \in [1, 900)$.

(2.2) For $\frac{\mathcal{D}_1^{p_h}+1}{2048} = \left\lfloor \frac{\mathcal{D}_1^{p_h}}{2048} \right\rfloor$, we have $\left\lfloor \frac{\mathcal{D}_1^{p_a}}{2048} \right\rfloor = \left\lfloor \frac{\mathcal{D}_1^{p_h}}{2048} \right\rfloor + 1$. At this point, it is only necessary to ensure that the following conditions hold:

$$\begin{aligned}&max\left\{1 - \left\lfloor \frac{t_2^{p_h} - t_1^{p_h}}{9} \right\rfloor, -99\right\} \cdot \left\lfloor \frac{\mathcal{D}_1^{p_h}}{2048} \right\rfloor \\ &- max\left\{1 - \left\lfloor \frac{t_2^{p_a} - t_1^{p_a}}{9} \right\rfloor, -99\right\} \cdot \left\lfloor \frac{\mathcal{D}_1^{p_a}}{2048} \right\rfloor \geq 0 \\ \Rightarrow\ &max\left\{1 - \left\lfloor \frac{t_2^{p_h} - t_1^{p_h}}{9} \right\rfloor, -99\right\} \cdot \left(\left\lfloor \frac{\mathcal{D}_1^{p_s}}{2048} \right\rfloor - 1\right) \\ &- max\left\{1 - \left\lfloor \frac{t_2^{p_a} - t_1^{p_a}}{9} \right\rfloor, -99\right\} \cdot \left\lfloor \frac{\mathcal{D}_1^{p_a}}{2048} \right\rfloor \geq 0.\end{aligned}$$

Since $max\left\{1 - \left\lfloor \frac{t_2^{p_a} - t_1^{p_a}}{9} \right\rfloor, -99\right\} < 0$, it follows that:

$$\begin{aligned}&\frac{max\left\{1 - \left\lfloor \frac{t_2^{p_h} - t_1^{p_h}}{9} \right\rfloor, -99\right\}}{max\left\{1 - \left\lfloor \frac{t_2^{p_a} - t_1^{p_a}}{9} \right\rfloor, -99\right\}} \leq \frac{\left\lfloor \frac{\mathcal{D}_1^{p_a}}{2048} \right\rfloor}{\left\lfloor \frac{\mathcal{D}_1^{p_s}}{2048} \right\rfloor - 1} \\ \Rightarrow\ &\left\lfloor \frac{max\left\{1 - \left\lfloor \frac{t_2^{p_h} - t_1^{p_h}}{9} \right\rfloor, -99\right\}}{max\left\{1 - \left\lfloor \frac{t_2^{p_a} - t_1^{p_a}}{9} \right\rfloor, -99\right\}} \right\rfloor \leq 1 \\ \Rightarrow\ &max\left\{1 - \left\lfloor \frac{t_2^{p_h} - t_1^{p_h}}{9} \right\rfloor, -99\right\} \\ &-max\left\{1 - \left\lfloor \frac{t_2^{p_a} - t_1^{p_a}}{9} \right\rfloor, -99\right\} \geq 0 \\ \Rightarrow\ &\begin{cases} 1 - \left\lfloor \frac{t_2^{p_h} - t_1^{p_h}}{9} \right\rfloor \geq 1 - \left\lfloor \frac{t_2^{p_a} - t_1^{p_a}}{9} \right\rfloor & (1) \\ 1 - \left\lfloor \frac{t_2^{p_a} - t_1^{p_a}}{9} \right\rfloor \geq -99 & (2) \end{cases}.\end{aligned} \tag{36}$$

To solve inequality (1), we have:

$$\left\lfloor \frac{t_2^{p_a} - t_1^{p_a} - (t_2^{p_h} - t_1^{p_h})}{9} \right\rfloor \geq 0. \tag{37}$$

To slove inequality (2), we have:

$$\left\lfloor \frac{t_2^{p_a} - t_1^{p_a}}{9} \right\rfloor \leq 100. \tag{38}$$

In order for a block $\mathcal{B}_2^{p_a}$ to be valid, it needs to be satisfied: $t_2^{p_a} - t_1^{p_a} \geq 1$. Thus we have:

$$t_2^{p_a} - t_1^{p_a} \in [1, 900). \tag{39}$$

Combining inequalities (37) and (39), if $i = 2$ and $\frac{\mathcal{D}_1^{p_h}+1}{2048} = \left\lfloor \frac{\mathcal{D}_1^{p_h}}{2048} \right\rfloor$, SUUM attack succeeds if and only if the following conditions hold: $\left\lfloor \frac{t_2^{p_h} - t_1^{p_h} - \left(t_2^{p_a} - t_1^{p_a}\right)}{9} \right\rfloor \geq 0$ and $t_2^{p_a} - t_1^{p_a} \in [1, 900)$.

(3) For $i \geq 3$, the proof process is analogous to the case where $i = 2$.

Appendix H. Proof of Theorem 7

To minimize the risk posed by the SUUM attack, which corresponds to minimizing the block difficulty growth rate, according to Theorem 6, we proceed as follows:

(1) For $i = 1$, we take the minimum value within the condition $\left\lfloor \frac{t_i^{p_h} - t_{i-1}^{p_a}}{9} \right\rfloor \in [1, +\infty)$, i.e., $\left\lfloor \frac{t_1^{p_h} - t_0^{p_a}}{9} \right\rfloor = 1$.

(2) For $i \geq 2$, we take the minimum value within the condition $\left\lfloor \frac{t_i^{p_h} - t_{i-1}^{p_h} - \left(t_i^{p_a} - t_{i-1}^{p_a}\right)}{9} \right\rfloor \in [1, +\infty)$, i.e., $\left\lfloor \frac{t_i^{p_h} - t_{i-1}^{p_h} - \left(t_i^{p_a} - t_{i-1}^{p_a}\right)}{9} \right\rfloor = 1$.

By integrating these two cases, we can prove the theorem.

Appendix I.
Proof of Theorem 8

Based on the analysis of state transitions in the Deployment State, Downgrade State, and Attack State of SUUM presented in Section 5.2, we calculate the absolute and relative shares of coin-base rewards for both the SUUM adversary and honest participants.

The absolute share of coin-base rewards for the SUUM adversary $\mathcal{P}_{\mathcal{A}}$ is:

$$\begin{aligned}\mathbb{R}_{\mathcal{P}_{\mathcal{A}}}^{SUUM} &= \mathbb{P}\left(Deploy\right)\cdot\mathbb{P}_{\mathcal{A}}\left(any\ t\right)\\ &+\mathbb{P}\left(Downgrade\right)\cdot\mathbb{P}_{\mathcal{A}}\left(any\ t\right)\\ &+\mathbb{P}\left(Attack\ \ i,i\geq 1\right)\cdot\mathbb{P}_{\mathcal{A}}\left(any\ t\right)\\ &=\begin{pmatrix}P\left(Deploy\right)+P\left(Downgrade\right)\\ +P\left(Attack\ \ i,i\geq 1\right)\end{pmatrix}\cdot\mathbb{P}_{\mathcal{A}}\left(any\ t\right)\\ &=\mathbb{P}_{\mathcal{A}}\left(any\ t\right).\end{aligned} \tag{40}$$

Therefore, the absolute share of coin-base rewards for the SUUM adversary $\mathcal{P}_{\mathcal{A}}$ remains unchanged.

The absolute share of coin-base rewards for honest participant $\mathcal{P}_{\mathcal{H}}$ is:

$$\begin{aligned}\mathbb{R}_{\mathcal{P}_{\mathcal{H}}}^{SUUM} &= \mathbb{P}\left(Deploy\right)\cdot\mathcal{P}_{\mathcal{H}}\left(t<9\right)\\ &+\mathbb{P}\left(Deploy\right)\cdot\mathbb{P}_{\mathcal{H}}\left(t\geq 9\right)\\ &+\mathbb{P}\left(Downgrade\right)\cdot\mathbb{P}_{\mathcal{H}}\left(t<9\right)\\ &+\mathbb{P}\left(Downgrade\right)\cdot\mathbb{P}_{\mathcal{H}}\left(t\geq 9\right)\\ &+\mathbb{P}\left(Attack\ \ i+1,i\geq 1\right)\cdot\mathcal{P}_{\mathcal{H}}\left(any\ t\right)\\ &-\mathbb{P}\left(Downgrade\right)\cdot\mathbb{P}_{\mathcal{A}}\left(any\ t\right)\\ &-\mathbb{P}\left(Attack\ \ i,i\geq 1\right)\cdot\mathcal{P}_{\mathcal{A}}\left(any\ t\right)\\ &=\begin{pmatrix}P\left(Deploy\right)+P\left(Downgrade\right)\\ +P\left(Attack\ \ i,i\geq 1\right)\end{pmatrix}\cdot\mathcal{P}_{\mathcal{H}}\left(any\ t\right)\\ &-\mathbb{P}\left(Attack\ \ 1\right)\cdot\mathcal{P}_{\mathcal{H}}\left(any\ t\right)\\ &-\mathbb{P}\left(Downgrade\right)\cdot\mathbb{P}_{\mathcal{A}}\left(any\ t\right)\\ &-\mathbb{P}\left(Attack\ \ i,i\geq 1\right)\cdot\mathcal{P}_{\mathcal{A}}\left(any\ t\right)\\ &=\mathcal{P}_{\mathcal{H}}\left(any\ t\right)-\mathbb{P}\left(Attack\ \ 1\right)\\ &-\mathbb{P}\left(Downgrade\right)\cdot\mathbb{P}_{\mathcal{A}}\left(any\ t\right)\\ &-\mathbb{P}\left(Attack\ \ i+1,i\geq 1\right)\cdot\mathcal{P}_{\mathcal{A}}\left(any\ t\right)\\ &<\mathcal{P}_{\mathcal{H}}\left(any\ t\right).\end{aligned} \tag{41}$$

Therefore, the absolute share of coin-base rewards for honest participants decreases.

The relative share of coin-base rewards for the SUUM adversary $\mathcal{P}_{\mathcal{A}}$ is:

$$\begin{aligned}\mathbb{E}\left[\mathbb{R}_{\mathcal{P}_{\mathcal{A}}}^{SUUM}\right] &= \frac{\mathbb{R}_{\mathcal{P}_{\mathcal{A}}}^{SUUM}}{\mathbb{R}_{\mathcal{P}_{\mathcal{A}}}^{SUUM}+\mathbb{R}_{\mathcal{P}_{\mathcal{H}}}^{SUUM}}\\ &=\frac{\mathbb{P}_{\mathcal{A}}\left(any\ t\right)}{\begin{pmatrix}\mathbb{P}_{\mathcal{A}}\left(any\ t\right)+\mathcal{P}_{\mathcal{H}}\left(any\ t\right)-P\left(Attack\ \ 1\right)\\ -P\left(Downgrade\right)\cdot\mathbb{P}_{\mathcal{A}}\left(any\ t\right)\\ +P\left(Attack\ \ i+1,i\geq 1\right)\cdot\mathcal{P}_{\mathcal{A}}\left(any\ t\right)\end{pmatrix}}\\ &>\frac{\mathbb{P}_{\mathcal{A}}\left(any\ t\right)}{\mathbb{P}_{\mathcal{A}}\left(any\ t\right)+\mathcal{P}_{\mathcal{H}}\left(any\ t\right)}\\ &=\mathbb{P}_{\mathcal{A}}\left(any\ t\right).\end{aligned} \tag{42}$$

Therefore, the expected relative share of coin-base rewards for the SUUM adversary $\mathcal{P}_{\mathcal{A}}$ increases.

The expected relative share of coin-base rewards for honest participant $\mathcal{P}_{\mathcal{H}}$ is:

$$\begin{aligned}\mathbb{E}\left[\mathbb{R}_{\mathcal{P}_{\mathcal{H}}}^{SUUM}\right] &= \frac{\mathbb{R}_{\mathcal{P}_{\mathcal{H}}}^{SUUM}}{\mathbb{R}_{\mathcal{P}_{\mathcal{A}}}^{SUUM}+\mathbb{R}_{\mathcal{P}_{\mathcal{H}}}^{SUUM}}\\ &=\frac{\begin{pmatrix}\mathbb{P}_{\mathcal{H}}\left(any\ t\right)-P\left(Attack\ \ 1\right)\\ -P\left(Downgrade\right)\cdot\mathbb{P}_{\mathcal{A}}\left(any\ t\right)\\ +P\left(Attack\ \ i+1,i\geq 1\right)\cdot\mathcal{P}_{\mathcal{A}}\left(any\ t\right)\end{pmatrix}}{\begin{pmatrix}\mathbb{P}_{\mathcal{A}}\left(any\ t\right)+\mathcal{P}_{\mathcal{H}}\left(any\ t\right)-P\left(Attack\ \ 1\right)\\ -P\left(Downgrade\right)\cdot\mathbb{P}_{\mathcal{A}}\left(any\ t\right)\\ +P\left(Attack\ \ i+1,i\geq 1\right)\cdot\mathcal{P}_{\mathcal{A}}\left(any\ t\right)\end{pmatrix}}\\ &<\frac{\mathcal{P}_{\mathcal{H}}\left(any\ t\right)}{\mathbb{P}_{\mathcal{A}}\left(any\ t\right)+\mathcal{P}_{\mathcal{H}}\left(any\ t\right)}\\ &=\mathcal{P}_{\mathcal{H}}\left(any\ t\right).\end{aligned} \tag{43}$$

Therefore, the expected relative share of coin-base rewards for honest participants $\mathcal{P}_{\mathcal{H}}$ decreases.

Appendix J.
Proof of Theorem 9

Firstly, it has been proved in the literature [39] that RUM outperforms honest mining. This provides us with a starting point for our subsequent comparisons.

Next, let's focus on the comparison between UUM and RUM. When we look at their state space and transition processes, we can find that there is a key difference in the conditions that trigger the transition from the deployment state to the attack state.

For RUM, it transitions from the deployment state to the attack state when an honest participant finds a new block whose timestamp differs from the timestamp of the previous block in the main chain by a certain value. Specifically, this value needs to be greater than or equal to 9 and less than 18, which we can denote as $\mathcal{P}_{\mathcal{H}}\left(9\leq t<18\right)$. In a practical blockchain scenario, this means that only when the time interval between the generation of consecutive blocks by honest participants falls within this specific range will RUM enter the attack state.

In contrast, for UUM, it transitions from the deployment state to the attack state when an honest participant finds

a new block whose timestamp differs from the previous block's timestamp in the main chain by a value greater than or equal to 9, denoted as $\mathcal{P}_{\mathcal{H}}(t \geq 9)$. For example, if we imagine the blockchain as a sequence of events over time, UUM is more likely to start an attack as it has a broader range of time differences that can trigger the attack state compared to RUM.

This difference in the transition conditions leads to a higher steady-state probability of being in the attack state for UUM compared to RUM. And as a result, UUM adversaries can obtain higher rewards than RUM adversaries.

Similarly, when we consider the comparison between SUUM and UUM, the approach to proving that SUUM outperforms UUM follows a similar logic. SUUM expands the adversary's state space in a significant way. In the attacking state, adversaries in SUUM can release held-back blocks with carefully selected timestamps. Let's say, they can choose the most opportune moments to release these blocks to maximize their impact on the blockchain's operation and ultimately obtain higher rewards than UUM adversaries. This is somewhat like having more strategic options in a game, which gives SUUM an advantage over UUM.

Appendix K. Proof of Theorem 10

We discuss the difficulty risks posed by each of the three types of attacks as follows:

(1) First, we analyze the difficulty risk posed by the RUM attack. Recalling Theorems 2 and 3 from Section 3, we understand that each successful execution of a RUM attack increases the difficulty risk of the blockchain by 1. We use the notation $Attack \overset{\mathcal{P}_{\mathcal{A}}}{\Rightarrow}{}^{any\ t} Deploy$ to represent this situation. That is, every time a RUM adversary successfully executes an attack and triggers the state change from the attack state to the deployment state, it leads to an increment in the blockchain's difficulty risk. Consequently, the increased risk associated with the RUM attack can be quantified as the number of successful attacks by a RUM adversary, denoted by $\sum \left(Attack \overset{\mathcal{P}_{\mathcal{A}}}{\Rightarrow}{}^{any\ t} Deploy \right)$.

(2) The proof of UUM follows a methodology analogous to that of RUM.

(3) Finally, we analyze the difficulty risk posed by SUUM attack. Recalling Theorems 6 and 7 from Section 5, we understand that each successful execution of a SUUM attack also increases the difficulty risk of the blockchain by 1. This increase in risk is associated with two specific types of state transitions, represented by $Downgrade \overset{\mathcal{P}_{\mathcal{A}}}{\Rightarrow}{}^{any\ t} Deploy$ and $Attack\ 1 \overset{\mathcal{P}_{\mathcal{H}}}{\Rightarrow}{}^{any\ t} Deploy$. In other words, when an SUUM adversary successfully conducts an attack and makes the state change happen in either of these two ways, it adds to the overall risk related to the blockchain's difficulty. Therefore, the increased risk associated with SUUM attack can be quantified as the number of successful attacks by a SUUM adversary, denoted by $\sum \begin{pmatrix} Downgrade \overset{\mathcal{P}_{\mathcal{A}}}{\Rightarrow}{}^{any\ t} Deploy \\ +Attack\ 1 \overset{\mathcal{P}_{\mathcal{H}}}{\Rightarrow}{}^{any\ t} Deploy \end{pmatrix}$.

Appendix L. Proof of Theorem 11

We discuss the forking rate posed by each of the three types of attacks and honest mining as follows:

(1) For honest mining, all participants adhere to the protocol and engage in honest mining practices. Under such circumstances, no participant undertakes malicious actions deliberately intended to cause forks. Consequently, the deliberate forking rate induced by honest mining, denoted as $\mathbb{FR}^{HM} = 0$.

(2) For the RUM attack, the adversary executes the RUM attack while other honest participants continue to mine honestly. Under such circumstances, revisiting the RUM attack methodology outlined in Section 3, we understand that the deliberate forking of the blockchain occurs as a result of a successful RUM attack by the adversary. A successful RUM attack is contingent upon the RUM adversary, who is in the attack state, successfully generating the next block, with the timestamp difference between it and its parent block being less than 9. Therefore, the deliberate forking rate induced by the RUM attack, denoted as $\mathbb{FR}^{RUM} = \mathbb{P}^{RUM}_{Attack} \cdot \mathcal{P}_{\mathcal{A}}(t < 9)$.

(3) For the UUM attack, the adversary executes the UUM attack while other honest participants continue to mine honestly. Under such circumstances, revisiting the UUM attack methodology outlined in Section 4, we understand that the deliberate forking of the blockchain arises due to a successful UUM attack by the adversary. A successful UUM attack occurs if and only if the UUM adversary, who is in the attack state, successfully generates the next block and strategically manipulates its timestamp such that the adversary's block in the fork competition is preferentially selected by other honest participants over the honest chain. Therefore, the deliberate forking rate induced by the UUM attack, denoted as $\mathbb{FR}^{UUM} = \mathbb{P}^{UUM}_{Attack} \cdot \mathcal{P}_{\mathcal{A}}(any\ t)$.

(4) For SUUM attack, the adversary executes the SUUM attack while other honest participants continue to mine honestly. In such a scenario, revisiting the SUUM attack methodology outlined in Section 5, we can identify two scenarios that lead to deliberate forking of the blockchain. The first scenario occurs when the blockchain topology is in the attack state, where the withholding of blocks by the SUUM adversary will inevitably cause a blockchain fork. The corresponding probability for this scenario is $\sum_{i=0}^{n} \mathbb{P}^{SUUM}_{Attack\ i}$. The second scenario is when the SUUM adversary, who is in the downgrade state, successfully generates the next block and strategically manipulates its timestamp such that this block is preferentially selected, leading to deliberate forking of the blockchain. The corresponding

TABLE 2. SUUM STATE TRANSITION.

State	Destination	Transition	Probability	Reward	
				$\mathcal{P}_{\mathcal{H}}$	$\mathcal{P}_{\mathcal{S}}$
Deployment	Attack 1	$\mathcal{P}_{\mathcal{A}}$ $any\ t$	$\mathbb{P}_{\mathcal{A}}(any\ t)$	0	1
Deployment	Deployment	$\mathcal{P}_{\mathcal{H}}$ $t<9$	$\mathbb{P}_{\mathcal{H}}(t<9)$	1	0
Deployment	Downgrade	$\mathcal{P}_{\mathcal{H}}$ $t\geq 9$	$\mathbb{P}_{\mathcal{H}}(t\geq 9)$	1	0
Downgrade	Deployment	$\mathcal{P}_{\mathcal{H}}$ $t<9$	$\mathbb{P}_{\mathcal{H}}(t<9)$	1	0
Downgrade	Downgrade	$\mathcal{P}_{\mathcal{H}}$ $t\geq 9$	$\mathbb{P}_{\mathcal{H}}(t\geq 9)$	1	0
Downgrade	Deployment	$\mathcal{P}_{\mathcal{A}}$ $any\ t$	$\mathbb{P}_{\mathcal{A}}(any\ t)$	-1	1
Attack 1	Deployment	$\mathcal{P}_{\mathcal{H}}$ $any\ t$	$\mathbb{P}_{\mathcal{H}}(any\ t)$	0	0
Attack i, $i\geq 1$	Attack $i+1$	$\mathcal{P}_{\mathcal{A}}$ $any\ t$	$\mathbb{P}_{\mathcal{A}}(any\ t)$	-1	1
Attack $i+1$, $i\geq 1$	Attack i	$\mathcal{P}_{\mathcal{H}}$ $any\ t$	$\mathbb{P}_{\mathcal{H}}(any\ t)$	1	0

probability for this scenario is $\mathbb{P}^{SUUM}_{Downgrade}\cdot\mathcal{P}_{\mathcal{A}}(any\ t)$. Therefore, the deliberate forking rate induced by the SUUM attack, denoted as $\mathbb{FR}^{SUUM}$, is given by the sum of the probabilities of these two scenarios:

$$\begin{aligned}\mathbb{FR}^{SUUM}&=\sum_{i=0}^{n}\mathbb{P}^{SUUM}_{Attack\ i}+\mathbb{P}^{SUUM}_{Downgrade}\cdot\mathcal{P}_{\mathcal{A}}\left(any\ t\right)\\&=\mathbb{P}^{SUUM}_{Attack}+\mathbb{P}^{SUUM}_{Downgrade}\cdot\mathcal{P}_{\mathcal{A}}(any\ t).\end{aligned}\tag{44}$$

Next, we proceed to compare the magnitudes of the forking rates arising from these four scenarios. On the one hand, according to Theorem 9, for adversaries possessing equivalent computational power, the steady-state probabilities of being in the attack state for SUUM, UUM, and RUM decrease sequentially, i.e.,

$$\mathbb{P}^{SUUM}_{Attack}>\mathbb{P}^{UUM}_{Attack}>\mathbb{P}^{RUM}_{Attack}.\tag{45}$$

On the other hand, it is evident that

$$1>\mathcal{P}_{\mathcal{A}}\left(any\ t\right)>\mathcal{P}_{\mathcal{A}}(t<9).\tag{46}$$

Therefore, we conclude that

$$\begin{aligned}&\mathbb{P}^{SUUM}_{Attack}>\mathbb{P}^{UUM}_{Attack}\cdot\mathcal{P}_{\mathcal{A}}\left(any\ t\right)>\mathbb{P}^{RUM}_{Attack}\cdot\mathcal{P}_{\mathcal{A}}\left(t<9\right)\\\Rightarrow&\mathbb{FR}^{SUUM}>\mathbb{FR}^{UUM}>\mathbb{FR}^{RUM}>\mathbb{FR}^{HM}.\end{aligned}\tag{47}$$

Appendix M. Proof of Theorem 12

We know that the relationship between $target$ and the difficulty $\mathcal{D}$ can be expressed as:

$$target=\frac{max\_target}{\mathcal{D}}.\tag{48}$$

Where, max_target is a constant, and for Ethereum 1.x, its value is 2^{256}.

The probability $P_{\mathcal{A}}$ for adversary $\mathcal{A}$ to find the next block can be calculated as follows:

$$\mathcal{P}_{\mathcal{A}}=\mu_{\mathcal{A}}\cdot\frac{1}{target}.\tag{49}$$

Since $target$ is inversely proportional to the difficulty $\mathcal{D}$, we can replace target with an expression related to $\mathcal{D}$:

$$\mathcal{P}_{\mathcal{A}}=\mu_{\mathcal{A}}\cdot\frac{\mathcal{D}}{max\_target}.\tag{50}$$

1) For RUM attack, its cost can be expressed as:

$$\begin{aligned}\mathbb{AC}^{RUM}&=\mathcal{P}_{\mathcal{A}}^{\mathcal{B}_0^{p_h}}-\mathcal{P}_{\mathcal{A}}^{\mathcal{B}_1^{p_h}}\\&=\mu_{\mathcal{A}}\cdot\frac{\mathcal{D}_0^{p_h}}{max\_target}-\mu_{\mathcal{A}}\cdot\frac{\mathcal{D}_1^{p_h}}{max\_target}\\&=\mu_{\mathcal{A}}\cdot\frac{(\mathcal{D}_0^{p_h}-\mathcal{D}_1^{p_h})}{max\_target}.\end{aligned}\tag{51}$$

According to [39], the initiation condition for the RUM attack is $\left\lfloor\frac{t_1^{p_h}-t_0^{p_h}}{9}\right\rfloor\in[1,9)$, i.e., $\mathcal{D}_0^{p_h}=\mathcal{D}_1^{p_h}$. Therefore, we have:

$$\mathbb{AC}^{RUM}=0.\tag{52}$$

2) For UUM and/or SUUM attacks, their cost can be expressed as:

$$\begin{aligned}\mathbb{AC}^{UUM}&=\mathbb{AC}^{SUUM}=\mathcal{P}_{\mathcal{A}}^{\mathcal{B}_0^{p_h}}-\mathcal{P}_{\mathcal{A}}^{\mathcal{B}_1^{p_h}}\\&=\mu_{\mathcal{A}}\cdot\frac{\mathcal{D}_0^{p_h}}{max\_target}-\mu_{\mathcal{A}}\cdot\frac{\mathcal{D}_1^{p_h}}{max\_target}\\&=\mu_{\mathcal{A}}\cdot\frac{(\mathcal{D}_0^{p_h}-\mathcal{D}_1^{p_h})}{max\_target}.\end{aligned}\tag{53}$$

According to Theorem 1 and Theorem 6, the initiation condition for UUM and SUUM attacks is $\left\lfloor\frac{t_1^{p_h}-t_0^{p_h}}{9}\right\rfloor\in[9,+\infty)$, i.e., $\mathcal{D}_0^{p_h}-\mathcal{D}_1^{p_h}\geq 1$. Therefore, we have:

$$\begin{aligned}\mathbb{AC}^{UUM}=\mathbb{AC}^{SUUM}&=\mu_{\mathcal{A}}\cdot\frac{(\mathcal{D}_0^{p_h}-\mathcal{D}_1^{p_h})}{max\_target}\\&>\mu_{\mathcal{A}}\cdot\frac{1}{max\_target}\\&>0.\end{aligned}\tag{54}$$

Taking Ethereum 1.x as an example, by substituting max_target with 2^{256}, we obtain:

$$\mathcal{P}_{\mathcal{A}}=\mu_{\mathcal{A}}\cdot\frac{\mathcal{D}}{2^{256}}.\tag{55}$$

Thus, the probability $\mathcal{P}_{\mathcal{A}}$ for an adversary $\mathcal{A}$, possessing a power ratio of $\mu_{\mathcal{A}}$, to find the next block is the proportion of its power to the total power, multiplied by the ratio of the block difficulty $\mathcal{D}$ to the maximum target value.

It is noteworthy that, upon reviewing historical data from Ethereum 1.x, we observe that the maximum difference in difficulty between two consecutive blocks is

approximately $8.86441324 \times 10^{12} \approx 2^{39.65}$. Therefore, we can derive:

$$\begin{aligned}\mathbb{AC}^{UUM} = \mathbb{AC}^{SUUM} &\leq \mu_{\mathcal{A}} \cdot \frac{max\left\{\mathcal{D}_0^{p_h} - \mathcal{D}_1^{p_h}\right\}}{2^{256}} \\ &\leq \mu_{\mathcal{A}} \cdot \frac{2^{39.65}}{2^{256}} \approx \mu_{\mathcal{A}} \cdot \frac{1}{2^{216.35}} \\ &\approx 0.\end{aligned} \tag{56}$$

Appendix N. Mitigations

N.1. Adjust the Difficulty Adjustment Algorithm

The difficulty adjustment algorithm of Ethereum 1.x is at risk of being exploited in the face of Uncle Maker attacks. Adversaries can manipulate the timestamp to influence the block difficulty and thus gain an unfair advantage. Therefore, it is necessary to design a more robust and manipulation-resistant difficulty adjustment algorithm. For example, more factors can be considered to determine the block difficulty. Instead of relying solely on the timestamp and the difficulty of the parent block, dynamic factors such as the actual computational power distribution of the network and the number of transactions can also be incorporated. In this way, it becomes difficult for adversaries to simply manipulate the timestamp to reduce their attack difficulty, thereby increasing the cost and difficulty of the attack and reducing the probability of the attack occurring.

One possible improvement direction is to adopt a moving average-based difficulty adjustment method, which takes into account the generation time and difficulty of multiple past blocks instead of just the current block and its parent block. This can smooth the difficulty adjustment process and reduce the exploitability of short-term fluctuations by adversaries. Additionally, upper and lower limits for difficulty adjustment can be set to prevent adversaries from causing significant drops or rises in difficulty through extreme timestamp manipulations, thereby maintaining the stability and predictability of the blockchain difficulty.

N.2. Strengthen Timestamp Verification

Given that the Uncle Maker attack is highly reliant on the manipulation of timestamps, strengthening the timestamp verification mechanism is one of the key measures to mitigate such attacks. A more rigorous and reliable source of timestamps can be established. For instance, a network of distributed timestamp servers can be employed to ensure that all nodes can obtain consistent and accurate time information. Simultaneously, stricter limitations on the range of block timestamps should be imposed to prevent adversaries from arbitrarily setting unreasonable timestamps. For example, it can be stipulated that the block timestamp must fall within a reasonable interval, be in line with the average block generation time of the network, and possess reasonable continuity with the timestamps of preceding blocks.

A timestamp consensus algorithm can be introduced, mandating that multiple nodes perform consensus verification on the block timestamp. Only when a certain proportion of mining power approve can the timestamp be regarded as valid. Additionally, for blocks with abnormal timestamps, more in-depth examination and verification can be carried out. Even the block can be temporarily rejected until the legality of its timestamp is ascertained. This can effectively prevent adversaries from launching Uncle Maker attacks by manipulating timestamps and enhance the security of the blockchain system.

N.3. Introduce an Economic Penalty Mechanism

To further deter adversaries, an economic penalty mechanism can be introduced. For nodes or participants detected to be involved in the Uncle Maker attack, in addition to confiscating the improper gains obtained through the attack, additional economic penalties can be imposed on them, such as deducting a certain proportion of collateral assets (if applicable) or future mining rewards. This can increase the cost of the attack and make adversaries more cautious when considering launching an attack.

A reporting mechanism can be established to encourage other nodes to report suspicious attack behaviors. For nodes that report successfully, a certain reward can be given, and the source of the reward can be the fines imposed on the adversaries. In this way, a community supervision mechanism can be formed to jointly maintain the security and fairness of the blockchain. At the same time, the attack behaviors and penalty situations should be regularly publicized to serve as a warning and reduce the number of potential adversaries.

N.4. Improve Network Monitoring and Early Warning Capabilities

Real-time monitoring of the operating status of the blockchain network and timely detection of abnormal block generation patterns and timestamp behaviors are crucial for quickly responding to and mitigating Uncle Maker attacks. Specialized monitoring tools and algorithms can be deployed to perform real-time monitoring and analysis of key indicators such as block timestamps, difficulty changes, and uncle block generation in the network. Once an abnormal situation is detected, an early warning signal should be issued in a timely manner to notify network participants to take corresponding measures, such as suspending the acceptance of suspicious blocks and activating the emergency response mechanism.

Utilize machine learning and artificial intelligence technologies to conduct in-depth analysis of network data and establish prediction models for attack behaviors. Through learning from historical data, patterns and trends that may indicate the imminent occurrence of Uncle Maker attacks can be identified, and preventive measures can be taken in advance. For example, if it is found that the block timestamp setting pattern of a certain node or participant is significantly different from normal behavior and is accompanied

by an abnormal increase in the frequency of uncle block generation, the system can automatically mark the node as a suspicious object and strengthen the monitoring of its subsequent behaviors. At the same time, the detected abnormal situations should be shared with the entire blockchain community to promote the community to jointly respond to attack threats and improve the security and stability of the entire network.