Prrr: Personal Random Rewards for Blockchain Reporting

Hongyin Chen
Technion
hongyin.chen.contact@gmail.com

Yubin Ke
Peking University
keyubin097@gmail.com

Xiaotie Deng
Peking University
xiaotie@pku.edu.cn

Ittay Eyal
Technion
ittay@technion.ac.il

***Abstract*—Smart contracts, the stateful programs running on blockchains, often rely on *reports*. *Publishers* are paid to publish these reports on the blockchain. Designing protocols that incentivize timely reporting is the prevalent *reporting problem*. But existing solutions face a security-performance trade-off: Relying on a small set of trusted publishers introduces centralization risks, while allowing open publication results in an excessive number of reports on the blockchain. We identify the root cause of this trade-off to be the standard symmetric reward design, which treats all reports equally. We prove that no symmetric-reward mechanism can overcome the trade-off.**

We present *Personal Random Rewards for Reporting* (*Prrr*), a protocol that assigns random heterogeneous values to reports. We call this novel mechanism-design concept *Ex-Ante Synthetic Asymmetry*. To the best of our knowledge, Prrr is the first game-theoretic mechanism (in any context) that deliberately forms participant asymmetry. Prrr employs a second-price-style settlement to allocate rewards, ensuring incentive compatibility and achieving both security and efficiency. Following the protocol constitutes a Subgame-Perfect Nash Equilibrium, robust against collusion and Sybil attacks. Prrr is applicable to numerous smart contracts that rely on timely reports.

1. Introduction

Blockchain *smart contracts* are stateful programs that facilitate the $170B Decentralized Finance ecosystem [1]. A wide variety of contracts require *reports* with external data for their functionality [2], [3], [4], [5], [6], [7], [8], [9], [10], [11], [12], [13], [14], [15], [16], [17]. In all cases, a smart contract requires a report to trigger its functionality. Principals called *publishers* should publish reports and blockchain operators, called *validators*, should include a report in the blockchain. But, as is the nature of decentralized systems, participants might not follow the protocol; they take the actions that maximize their profits. The smart contract therefore pays the publishers with cryptocurrency, and the challenge is to correctly incentivize the desired behavior. We call this the *Blockchain Reporting Problem*.

Previous work (§2) and a variety of deployed systems face this problem. Many pay a fixed reward to the publisher who generated the first report placed on chain. Others include multiple reports on chain and distribute a reward among them. However, this approach creates a security-performance trade-off: Relying on a small set of publishers introduces a single point of failure, as demonstrated in recent incidents [18], [19], [20]. Conversely, allowing anyone to publish reports results in intense competition. This erodes the participation incentive or leads to an excessive number of on-chain reports (even in protocols that reward just one report), as observed in liquidation events [14], [21], [22].

To analyze the reporting problem, we model (§3) a smart contract requesting one report. There are two types of rational participants: *publishers* and *blockchain validators*. Validators generate blocks that contain *transactions*. Publishers generate reports and send them to validators in transactions along with fees and bribes. The set of publishers is unbounded and unknown to the contract in advance. The validators choose which reports to include in the block.

Our goal is to design a reporting protocol that ensures *efficiency*, i.e., the number of included reports is upper bounded by a constant C; *decentralization*, i.e., participants have positive revenue, incentivizing their participation; *fairness*, i.e., participants are rewarded in proportion to their efforts, to avoid centralization; and *progress*, i.e., reports are included in a timely manner, preventing censorship.

We find that the standard symmetric mechanism design principle (§4) is the root cause of the security-performance trade-off of previous solutions. This commonly used principle treats all reports symmetrically, meaning any two reports are interchangeable. We prove that no symmetric mechanism can achieve our desired properties if the total number of generated reports exceeds the bound C (for any C).

We therefore propose *Ex-Ante Synthetic Asymmetry* (*EASA*, §5), a mechanism-design concept that induces ex-ante randomness to create controlled asymmetry. To the best of our knowledge, EASA is the first game-theoretic approach that deliberately forms participant asymmetry.

Based on EASA, we present *Personal Random Rewards for Reporting* (*Prrr*), a reporting protocol that assigns random heterogeneous values to reports with a specific random-value function. Prrr requires publishers to publish all reports without offering bribes to the validators. To incentivize both participant types to follow the protocol, Prrr requires the validator to include in its block the reports with the highest and second-highest random values. The contract then rewards the validator with an amount equal to the value of the second-highest report, and it rewards the publisher who published the highest-value report with a reward equal to the difference

between its value and the second-highest value. Security thus relies on a careful choice of the random-value function. It should ensure that publisher rewards are monotonic in the number of published reports and should ensure validators include transactions, even in the face of bribes. We provide two functions that satisfy these requirements.

As is typical in decentralized algorithms [23], [24], [25], [26], [27], the security of Prrr relies on its incentive compatibility. Prrr gives rise to a sequential game among the publishers and the validators (§6). Each step of the game consists of two phases. In the first phase, publishers publish reports and offer bribes to the validator. In the second phase, the validator selects a vector of reports to include in the block based on the received reports and bribes.

We analyze the game (§7) using backward induction and demonstrate that adhering to the Prrr protocol constitutes a Subgame-Perfect Nash Equilibrium. We further show the robustness of this equilibrium: First, collusion among publishers and Sybil attacks are not beneficial. Second, collusion between publishers and the validator in the current step does not yield higher utility for the colluding parties. Third, any deviation by validators strictly reduces their own utility (with one of our random-value functions). And finally, if a publisher's deviation impacts the revenue of other publishers, the deviating publisher's own revenue strictly decreases.

In all, our main contributions are: (1) Modeling of the reporting problem; (2) Proof that no symmetric mechanism satisfies our desired properties; (3) A novel mechanism-design concept that assigns heterogeneous values to reports via ex-ante randomness; (4) Prrr, a second-price-style reward allocation rule using the above concept; (5) Carefully chosen random-value functions for Prrr; and (6) Game-theoretic analysis showing that Prrr is a robust equilibrium.

Prrr is directly applicable to the numerous smart contracts that rely on reports. It enables, for the first time, decentralizing this central piece of the blockchain ecosystem.

2. Related Work

Although numerous works consider stochastic settings in mechanism design (see [28] and, e.g., [29], [30], [31]), to the best of our knowledge, we are the first to deliberately impose participant asymmetry to achieve desired incentives.

Many blockchain smart contracts rely on external data to maintain their functionality. It arises in a variety of applications like Rollups [2], [3], [4], [32], cross-chain protocols [5], [6], [7], [33], misbehavior reporting protocols [8], [9], [10], [11], [12], [13], [34], [35], [36], [37], and financial events [14], [15], [16], [17].

All these settings share the same essential feature: a smart contract requires *at least one participant* to publish a report in order to trigger a procedure. The smart contract allocates rewards to publishers and publishers compete for rewards when publishing such reports. Mechanisms without such competition fall outside our scope. For example, Proof-of-Work NFT [38] mints and other parallel minting schemes let each submitter independently create or claim value, so there is no race over the same report. Likewise, collaborative aggregation systems (e.g., Oracles [39], [40], [41], [42]) seek quorum or averaging rather than compete on publishing functionally-equivalent reports, and thus do not fit our model.

Most existing mechanisms for the blockchain reporting problem pay a fixed bounty to the first valid report. This winner-takes-all rule creates a security-performance trade-off. If a protocol designates a small set or even a single publisher, authority concentrates and a single point of failure emerges. This raises manipulation and censorship risk and has caused real outages in rollups [18], [19], [20]. Conversely, if the protocol allows any publisher to publish reports (e.g., liquidation protocols), publishers race to be first, leading to congestion and higher fees in practice [14], [21], [22].

This trade-off cannot be resolved by fair transaction ordering, which aims to sequence blockchain transactions based on the order of their publication [43], [44], [45], [46], [47]. A publisher with superior connectivity or computational power can consistently publish first, securing rewards and discouraging others, ultimately driving the system toward centralization.

Some works pursue decentralized report *generation* in rollups [3], [4], but the reporting problem is about *publication and inclusion* on chain, which these designs do not address.

Other efforts target reporting within particular domains. ZkFair [48] rewards multiple publishers for publishing the same claim, but it forgoes efficiency, as every report must be recorded on chain. Chainlink [40] adopts a watchdog-priority scheme for price-bias alerts: Publishers receive sequential opportunities to publish an alert, and if a higher-priority node does not publish, the chance passes to the next. This still relies on a predefined set of publishers.

3. The Reporting Problem

We present the participants of the reporting problem (§3.1), the blockchain model (§3.2), how the system progresses (§3.3), how reports are distributed (§3.4), and the goal of the reporting protocol (§3.5).

3.1. Participants

The system comprises two sets of participants: a set $\mathbf{P} = \{P_1, P_2, \ldots, P_n\}$ of *Publishers* and a set $\mathbf{V}$ of blockchain *validators*. Publisher P_j generates *reports*, each with a fixed *cost* c_j. She can generate a limited number of reports within a time interval t, which we denote by $N_j^{\max}(t)$. Note that we do not predefine the set of publishers and allow for public participation. The definition of sets is used solely for analytical purposes. Decentralized systems can enforce both rate-limiting and uniqueness of reports using two main approaches: Proof-of-Work (PoW) and Proof-of-Stake (PoS). In PoW, generating reports incurs computational costs. The system enforces rate-limiting through computational power and ensures uniqueness by requiring a nonce for each report [49], [50]. In PoS, generating reports

requires financial stake. The system enforces rate-limiting based on the amount of stake and ensures uniqueness by assigning a unique index to each report [36].

Each validator $V \in \mathbf{V}$ holds a public key *PK* and a private key *SK*. The public key is common knowledge and only she knows her private key. Publishers and validators are rational and aim to maximize their revenue.

3.2. Blockchain

The blockchain is a replicated state machine whose transitions are recorded as a ledger, a sequence of log entries called *transactions*. *Smart contracts* are stateful programs that run on the blockchain. They are triggered by transactions, execute arbitrary logic, and update their internal state. They can also receive and send virtual tokens (cryptocurrency). Validators execute a state machine replication (consensus) protocol, extending the list of blockchain transactions by appending batches of transactions called *blocks* to the blockchain. Validators take turns generating the blocks and appending them to the blockchain, forming a total order of blocks. We denote the block in position (index) k by B^k and the validator who generates it by V^k. We omit the details of the blockchain protocol and focus on the block contents.

Before V^k generates block B^k, an external source provides a random beacon pulse Q^k. No one has any information about this beacon before the appending of block B^{k-1} or can influence its randomness. And after the appending of block B^{k-1}, the beacon pulse is public and becomes common knowledge. Such randomness [51], [52] is already in use for blockchain mechanism design [53]. To allow flexibility in the use of randomness, validator V^k generates an additional unbiased pseudorandom string S^k from the beacon pulse Q^k. We do not impose restrictions on how S^k is generated; it may be identical to Q^k or derived with other methods chosen by the validator. Hereafter, we refer to S^k as the random string.

The k-th block is a tuple containing its index k, the random beacon pulse Q^k, the random string S^k, and a vector of transactions $\mathit{TX}^k = (tx_1, tx_2, \cdots)$, i.e., $B^k = (k, Q^k, S^k, \mathit{TX}^k)$. A reporting transaction *tx* is a pair of a data structure *report*, *Rpt*, and a non-negative value b representing the amount of fees the participant who generated the transaction is willing to pay for validators to include the transaction, i.e., (Rpt, b). For simplicity, we only consider reporting transactions in this paper, ignoring unrelated transactions. We refer to b as the *transaction fee bid* of transaction *tx*.

3.3. System Progress

A smart contract requires a report to maintain its functionality within a time window spanning T blocks. We call the time window an *epoch*. There can be either regularly repeated sequential epochs (e.g., Rollups [54], [55]) or a single epoch (e.g., Watchtowers [10]). In practical systems, a smart contract might handle multiple epochs simultaneously for independent processes. However, in this discussion, we focus on the scenario involving a single epoch, as repetitions are multiple independent instances of a single epoch.

Publishers put each report in a transaction and send it to validators. Validators maintain a local set of transactions. Upon receiving a transaction, a validator appends it to the set. When generating a block B^k, validator V^k selects transactions from her local set of transactions as the transaction list TX^k in the block B^k. If $tx \in \mathit{TX}^k$, we say that V^k includes *tx*.

Publishers can also directly bribe validators. This bribery enables publishers to influence the contents of blocks to maximize their rewards. Formally, a publisher P_j provides a bribe function to validators, which we denote by Bribe_{Pj}. It takes a block B as input and outputs the bribe amount $\mathit{Bribe}_{Pj}(B)$ for the validator if she generates the block. Practical systems like Flashbots [56] already facilitate such payments.

The system progresses in steps within an epoch, where each step corresponds to a block. Within each step, first publishers generate and send transactions and offer bribes to validators, and then validators generate a block and append it to the blockchain.

3.4. Reporting Protocol

The *reporting protocol* indicates how publishers publish reports, validators include reports, and the smart contract distributes rewards to them. Starting from a certain step within the epoch, participants execute the following protocol every step until the end of this epoch, which consists of three phases: *Publication*, *Inclusion*, and *Processing*.

3.4.1. Publication Phase. Each publisher P_j has generated a set of reports $\boldsymbol{\mathit{Rpt}}_j$. In this phase, each publisher P_j forms a set of transactions PubR_j from her reports $\boldsymbol{\mathit{Rpt}}_j$ and generates a bribe function Bribe_{Pj}. Then, publisher P_j publishes $(\mathit{PubR}_j, \mathit{Bribe}_{Pj})$ to validators.

3.4.2. Inclusion Phase. When a validator V^k starts to generate a block, she executes the inclusion phase of the reporting protocol. She first generates the random string S^k from the beacon pulse Q^k and selects a set of transactions TX^k from her local transaction set. Then, she takes Q^k, S^k and TX^k to create the block $B^k = (k, Q^k, S^k, \mathit{TX}^k)$ and appends it to the blockchain.

The number of validators is large, so the validator V^k in each block k within the epoch is distinct and has distinct interests.

3.4.3. Processing Phase. When a validator V^k appends a block B^k to the blockchain, the smart contract executes the processing phase of the reporting protocol. This phase takes the vector of transactions TX^k of block B^k and the random string S^k as input and outputs the payment for reports and the validator.

3.5. Goal

A protocol solves the reporting problem if it satisfies *decentralization*, *fairness*, *efficiency* and *progress*. We define these properties as follows.

We do not rely on a predifined set of publishers. Therefore, decentralization means publishers are motivated to join the system.

Definition 1 (Decentralization)**.** *A reporting protocol is* decentralized *if all participants have positive expected revenue ignoring the cost of generating reports.*

Fairness ensures that publishers with many reports cannot gain additional rewards by splitting their identity into multiple smaller publishers. It also ensures that multiple publishers cannot increase their rewards by pooling together as a single publisher.

Definition 2 (Fairness)**.** *A reporting protocol is* fair *if the expected revenue ignoring the cost of generating reports for each publisher is proportional to the number of reports she generates and is independent of how the total reports are distributed among publishers.*

Both decentralization and fairness are crucial to incentivize participants to join the system.

Efficiency indicates that there are not excessive on-chain reports which cause inefficiency in the underlying blockchain system:

Definition 3 (C-Efficiency)**.** *A reporting protocol is* efficient *if validators include at most a constant number C of reports in total in an epoch, regardless of how many reports publishers generate.*

Progress property ensures that the smart contract can process reports quickly, reflecting the censorship-resistance of the protocol.

Definition 4 (Progress)**.** *A reporting protocol satisfies* progress *if the validator includes at least one report at the first step where participants start executing the reporting protocol.*

A smart contract executes the processing phase as prescribed. However, publishers and validators might deviate from the protocol to maximize their revenue. Since we cannot force participants to follow the protocol, ensuring security requires that the protocol design makes deviations unprofitable. In other words, the protocol must be incentive-compatible, that is, following the protocol is an equilibrium.

4. Impossibility under Symmetric Design

In the reporting problem, all valid reports are functionally equivalent. This naturally leads to the consideration of *symmetric reporting protocols*, which treat all reports equally, independent of the publisher's identity. We show that no incentive-compatible symmetric reporting protocol can achieve all of efficiency, decentralization and progress.

Symmetric Reporting Protocol. Before formally defining symmetry, we introduce some notation.

If a publisher publishes a bid for a transaction, it is equivalent to offering an additional bribe contingent on the transaction's inclusion. Therefore, without loss of generality, we merge bids into bribes and consider only bribes in the following discussion. Correspondingly, we abstract away transactions and only consider reports in the publication and inclusion.

We now formalize the reward and bribe functions. Given a vector of reports $\boldsymbol{Rpt} = (Rpt_1, Rpt_2, \ldots, Rpt_l)$ and a random string S as input, the publisher reward function outputs a vector of rewards for each report, $R_{\mathsf{publisher}}(\boldsymbol{Rpt}, S) = (r_1, r_2, \ldots, r_l)$ and the validator reward function outputs a reward for the validator, $R_{\mathsf{validator}}(\boldsymbol{Rpt}, S) = r_V$. Given a vector of reports $\boldsymbol{Rpt}$ and a random string S as input, the bribe function of a publisher outputs a non-negative bribe amount $Bribe(\boldsymbol{Rpt}, S)$ for the validator.

A symmetric protocol should have symmetric reward for reports, which treats all reports equally. This means that replacing any report in the input vector with another report should not change the output of the reward function.

Definition 5 (Symmetric Reward)**.** *For any two vectors of reports of equal length, $\boldsymbol{Rpt}$ and $\boldsymbol{Rpt}'$, and for any random string S, the reward functions allocate rewards identically: $R_{publisher}(\boldsymbol{Rpt}, S) = R_{publisher}(\boldsymbol{Rpt}', S)$ and $R_{validator}(\boldsymbol{Rpt}, S) = R_{validator}(\boldsymbol{Rpt}', S)$.*

We aim to design a reporting protocol that excludes bribes (and bids). However, since we are proving an impossibility, we allow bribes in the protocol design to strengthen the result. We still impose symmetry constraints on such bribes since the protocol is symmetric. Note that transaction fee bids included in bribes do not break the symmetry, as the bid for a report is paid only when the report is included.

Definition 6 (Symmetric Bribery)**.** *For any two vectors of reports of equal length, $\boldsymbol{Rpt} = (Rpt_1, \ldots, Rpt_l)$ and $\boldsymbol{Rpt}' = (Rpt'_1, \ldots, Rpt'_l)$, and for any random string S, the bribe function of publisher P_j is identical if her reports are in the same positions in both vectors, that is, if $\forall 1 \leq i \leq l$: $Rpt_i \in \boldsymbol{Rpt}_{P_j} \vee Rpt'_i \in \boldsymbol{Rpt}_{P_j}$ implies $Rpt_i = Rpt'_i$, then $Bribe_{P_j}(\boldsymbol{Rpt}, S) = Bribe_{P_j}(\boldsymbol{Rpt}', S)$.*

Note that publishers can still choose to deviate to an asymmetric bribery function since the protocol cannot enforce their actions.

Symmetric Game. Assume there exists a symmetric reporting protocol satisfying efficiency, fairness, progress, and incentive compatibility. We aim to demonstrate that if any publisher has positive expected revenue, another publisher can deviate in a way that strictly increases their expected revenue. This contradicts incentive compatibility. To establish this, we first formalize the game-theoretic model of a general symmetric reporting protocol.

Since the protocol satisfies progress, the validator includes at least one report in the first step. There are two

types of deviations. First, a publisher may attempt to bribe the validator to exclude all reports in this step. Second, a publisher may deviate to increase her own expected revenue while still ensuring that the validator includes at least one report in this step. For our impossibility proof, we focus on constructing the second type of deviation. After the first step, no participant can obtain any further reward from the smart contract, regardless of their actions. Recall that we already assume that validators have independent interests (§3.4), the validator in the first step is only concerned with her revenue for that step.

Therefore, it suffices to construct the game in the first step with the second type of deviation. Publishers first publish their transactions and bribe functions, after which the validator selects which transactions to include. This interaction forms a multi-leader, single-follower Stackelberg game.

The actions of the participants are as follows. For any publisher P_j with a set of reports $\boldsymbol{Rpt}_{P_j}$, her action consists of selecting a subset of reports $\mathit{PubR}_{P_j} \subseteq \boldsymbol{Rpt}_{P_j}$ to publish, along with a bribe function Bribe_{P_j} offered to the validator. The validator's action is to select an ordered vector $\boldsymbol{IncR}$ of reports to include in the block. The validator chooses this vector from the union of all published reports $\bigcup_{1\le j\le n} \mathit{PubR}_{P_j}$, and determines the final set and order of reports included in the block.

Denote by $R(\boldsymbol{IncR}, \boldsymbol{Rpt}, S)$ the total reward allocated by the smart contract to the set of reports $\boldsymbol{Rpt}$, given the included report vector $\boldsymbol{IncR} = (\mathit{IncR}_1, \mathit{IncR}_2, \ldots)$ and random string S. Let $R_{\mathsf{publisher}}(i, \boldsymbol{Rpt}, S)$ be the reward assigned to the i-th report Rpt_i in $\boldsymbol{Rpt}$, that is, i-th element in the vector $R_{\mathsf{publisher}}(\boldsymbol{Rpt}, S)$. Then, R is the sum of the rewards for all reports in $\boldsymbol{IncR}$ that also belong to $\boldsymbol{Rpt}$:

$$R(\boldsymbol{IncR}, \boldsymbol{Rpt}, S) = \sum_{\substack{\mathit{IncR}_i \in \boldsymbol{IncR} \ \&\\ \mathit{IncR}_i \in \boldsymbol{Rpt}}} R_{\mathsf{publisher}}(i, \boldsymbol{Rpt}, S) \ .$$

We now define the revenue for each participant. Since the cost of generating reports for each publisher is fixed and independent of their actions within the game, we omit it from the revenue definition. Given the random string S, the action of a publisher P_j, $(\mathit{PubR}_{P_j}, \mathit{Bribe}_{P_j})$, and the validator's action $\boldsymbol{IncR}$, the revenue of P_j is the total reward from her included reports minus her bribe paid to the validator:

$$R_{P_j}(\mathit{PubR}_{P_j}, \mathit{Bribe}_{P_j}, \boldsymbol{IncR}, S) = R(\boldsymbol{IncR}, \mathit{PubR}_{P_j}, S) - \mathit{Bribe}_{P_j}(\boldsymbol{IncR}, S) \ . \quad (1)$$

The utility of a publisher P_j is her expected revenue over the randomness of S, denoted by $E_S[\cdot]$.

$$U_{P_j}(\mathit{PubR}_{P_j}, \mathit{Bribe}_{P_j}, \boldsymbol{IncR}) = E_S[R_{P_j}(\mathit{PubR}_{P_j}, \mathit{Bribe}_{P_j}, \boldsymbol{IncR}, S)] \ .$$

For the validator, given the vector of bribe functions of all publishers $\boldsymbol{Bribe} = (\mathit{Bribe}_{P1}, \ldots, \mathit{Bribe}_{Pn})$, her action $\boldsymbol{IncR}$ and the random string S, her revenue is the reward from the smart contract plus the sum of bribes from all publishers:

$$R_V(\boldsymbol{Bribe}, \boldsymbol{IncR}, S) = R_{\mathsf{validator}}(\boldsymbol{IncR}, S) + \sum_{1\le j\le n} \mathit{Bribe}_{P_j}(\boldsymbol{IncR}, S) \ . \quad (2)$$

Impossibility under Symmetry. We show that no symmetric reporting protocol can achieve all of our desired goals.

Theorem 1. *For any reporting protocol with symmetric rewards and symmetric bribery, if it satisfies incentive compatibility, efficiency (i.e., includes at most C reports), and progress, then whenever the total number of reports generated by all publishers exceeds C, all publishers have zero expected revenue.*

Intuitively, suppose the total number of reports generated by all publishers is $N > C$. By the fairness property, the expected revenue for each report is the same as in the case where N publishers each generate one report. Assume, for contradiction, that the expected revenue for each report is positive, so at least one publisher has positive revenue for some execution with a random string S. However, since $N > C$, at least one publisher must have no report included by the validator in this execution. We construct a deviation for this excluded publisher: She offers a small bribe to replace the report of a publisher with a positive revenue, capturing part of that positive revenue. This profitable deviation contradicts incentive compatibility.

Proof. Assume there exists an instance of the game in which the total number of reports generated by all publishers is $N > C$, and the expected revenue for at least one publisher is positive. By the fairness property, the expected revenue for each report is also positive and identical to the case where N publishers each generate one report, since the total number of reports is the same. Therefore, it suffices to show that if there are $N > C$ publishers, each generating one report, and the expected revenue for each report is positive, then the protocol cannot be incentive-compatible.

We prove by contradiction. It suffices to construct a deviation for any publisher and a corresponding best response for the validator such that the publisher's revenue increases and the validator's action is uniquely optimal. In other words, if the publisher deviates to the new action, the validator's only best response results in strictly higher revenue for the publisher. This contradicts the assumption that the protocol is incentive-compatible.

For any publisher P_j where $1 \le j \le n$, we denote her only report by Rpt_{P_j}. She should publish it, i.e., $\mathit{PubR}_{P_j} = \{\mathit{Rpt}_{P_j}\}$. Otherwise, she would have zero revenue, contradicting the assumption that the expected revenue for each generated report is positive.

Without loss of generality, we assume publisher P_1 has a positive expected revenue. It means that there exists a value of the random string, S^*, such that her revenue is positive, denoted by $R_1^* > 0$.

In this case, we denote the best response taken by the validator by $\boldsymbol{IncR}^* = (IncR_1^*, IncR_2^*, \ldots)$. Due to efficiency, the validator includes at most C reports. Therefore, at least one publisher has no report included by the validator in this case. Without loss of generality, we denote this publisher by P_2. So P_2 gets zero reward in this case:

$$R(\boldsymbol{IncR}^*, \{Rpt_{P2}\}, S^*) = 0 \ . \tag{3}$$

We first construct a deviation for publisher P_2 that increases her revenue and find the validator's best response following this deviation. Then, we calculate the revenue of publisher P_2 showing that the deviation is profitable for her.

(i) **Deviation and Validator's Best Response**.

We construct a deviation letting P_2 seize P_1's revenue R_1^* with the random string S^*.

Since $\boldsymbol{IncR}^*$ is the validator's best response with random string S^*, the validator has the maximal revenue for choosing $\boldsymbol{IncR}^*$:

$$\forall \boldsymbol{IncR} : R_V(\boldsymbol{Bribe}, \boldsymbol{IncR}, S^*) \leq R_V(\boldsymbol{Bribe}, \boldsymbol{IncR}^*, S^*) \ . \tag{4}$$

Denote by $\boldsymbol{IncR}'$ the report vector that is identical to the original validator's best response $\boldsymbol{IncR}^*$ but with P_1's report Rpt_{P1} replaced by P_2's previously unincluded report Rpt_{P2}.

Publisher P_2 constructs a new bribe function $Bribe'_{P2}$. Given the random string S^*, the new bribe function is identical to her original bribe function $Bribe_2$ for all possible included vectors, except for $\boldsymbol{IncR}'$. Publisher P_2 pays the bribe to $\boldsymbol{IncR}'$ which is the sum of the bribes that both P_1 and P_2 originally paid for $\boldsymbol{IncR}^*$, plus ($1/3$ of R_1^*):

$$Bribe'_{P2}(\boldsymbol{IncR}, S^*) = \begin{cases} Bribe_{P2}(\boldsymbol{IncR}^*, S^*) + \\ \quad Bribe_{P1}(\boldsymbol{IncR}^*, S^*) + R_1^*/3 & \text{if } \boldsymbol{IncR} = \boldsymbol{IncR}'; \\ Bribe_{P2}(\boldsymbol{IncR}, S^*) & \text{otherwise.} \end{cases} \tag{5}$$

Now, we analyze the validator's revenue. We denote by $\boldsymbol{Bribe}' = (Bribe_{P1}, Bribe'_{P2}, Bribe_{P3}, \ldots, Bribe_{Pn})$ the vector of all publishers' bribes of the deviation case. The validator's revenue for choosing the report vector $\boldsymbol{IncR}$ with the new bribe functions and the random string S^* is (Equation 2):

$$\begin{aligned} & R_V(\boldsymbol{Bribe}', \boldsymbol{IncR}', S^*) \\ =& R_{\text{validator}}(\boldsymbol{IncR}', S^*) + Bribe'_{P2}(\boldsymbol{IncR}', S^*) + \\ & \sum_{j \neq 2} Bribe_{Pj}(\boldsymbol{IncR}', S^*) \\ =& R_{\text{validator}}(\boldsymbol{IncR}', S^*) + Bribe_{P2}(\boldsymbol{IncR}^*, S^*) + R_1^*/3 + \\ & Bribe_{P1}(\boldsymbol{IncR}^*, S^*) + \sum_{j \neq 2} Bribe_{Pj}(\boldsymbol{IncR}', S^*) \ . \end{aligned} \tag{6}$$

Now we consider the validator's revenue for choosing any report vector $\boldsymbol{IncR}$ other than $\boldsymbol{IncR}'$. Given the new bribe functions and the random string S^*, the validator's revenue for choosing the report vector $\boldsymbol{IncR}$ is (Equation 2):

$$\begin{aligned} & R_V(\boldsymbol{Bribe}', \boldsymbol{IncR} \neq \boldsymbol{IncR}', S^*) \\ =& R_{\text{validator}}(\boldsymbol{IncR}, S^*) + Bribe'_{P2}(\boldsymbol{IncR}, S^*) + \\ & \sum_{j \neq 2} Bribe_{Pj}(\boldsymbol{IncR}, S^*) \\ \overset{Eq.5}{=}& R_{\text{validator}}(\boldsymbol{IncR}, S^*) + Bribe_{P2}(\boldsymbol{IncR}, S^*) + \\ & \sum_{j \neq 2} Bribe_{Pj}(\boldsymbol{IncR}, S^*) \\ =& R_V(\boldsymbol{Bribe}, \boldsymbol{IncR}, S^*) + \sum_{1 \leq j \leq n} Bribe_{Pj}(\boldsymbol{IncR}, S^*) \end{aligned}$$

This is exactly the revenue of the validator for choosing the original best response $\boldsymbol{IncR}$ to the original bribe functions $\boldsymbol{Bribe}$ and the random string S^*. Combining with Equation 4, we have that the revenue of the validator for choosing any report vector $\boldsymbol{IncR}$ other than $\boldsymbol{IncR}'$ with the new bribe function is upper bounded by the revenue for choosing the original best response $\boldsymbol{IncR}^*$ with the original bribe functions:

$$R_V(\boldsymbol{Bribe}', \boldsymbol{IncR} \neq \boldsymbol{IncR}', S^*) \leq R_V(\boldsymbol{Bribe}, \boldsymbol{IncR}^*, S^*) \ . \tag{7}$$

Combining the above inequality with Equations 2 and 6, we calculate the difference in the validator's revenue between choosing $\boldsymbol{IncR}'$ and any other report vector $\boldsymbol{IncR}$ with the deviation bribe functions $\boldsymbol{Bribe}'$ and the random string S^*:

$$\begin{aligned} & R_V(\boldsymbol{Bribe}', \boldsymbol{IncR}', S^*) - \\ & R_V(\boldsymbol{Bribe}', \boldsymbol{IncR} \neq \boldsymbol{IncR}', S^*) \\ \overset{Eq.7}{\geq}& R_V(\boldsymbol{Bribe}', \boldsymbol{IncR}', S^*) - R_V(\boldsymbol{Bribe}, \boldsymbol{IncR}^*, S^*) \\ \overset{Eq.2,6}{=}& (R_{\text{validator}}(\boldsymbol{IncR}', S^*) - R_{\text{validator}}(\boldsymbol{IncR}^*, S^*)) + \\ & Bribe_{P1}(\boldsymbol{IncR}^*, S^*) + R_1^*/3 + \\ & \sum_{j \neq 2} (Bribe_{Pj}(\boldsymbol{IncR}', S^*) - Bribe_{Pj}(\boldsymbol{IncR}^*, S^*)) \ . \end{aligned}$$

Since the only difference between $\boldsymbol{IncR}'$ and $\boldsymbol{IncR}^*$ is the inclusion of Rpt_{P2} instead of Rpt_{P1}, due to the symmetric reward property, we have $R_{\text{validator}}(\boldsymbol{IncR}', S^*) = R_{\text{validator}}(\boldsymbol{IncR}^*, S^*)$. Similarly, for any other publisher $P_j \neq P_1, P_2$, their reports in $\boldsymbol{IncR}^*$ and $\boldsymbol{IncR}'$ are identical. Due to symmetric bribery, we have $Bribe_{Pj}(\boldsymbol{IncR}', S^*) = Bribe_{Pj}(\boldsymbol{IncR}^*, S^*)$. Therefore, we can simplify the above difference to:

$$\begin{aligned} R_V(\boldsymbol{Bribe}', \boldsymbol{IncR}', S^*) - R_V(\boldsymbol{Bribe}', \boldsymbol{IncR} \neq \boldsymbol{IncR}', S^*) = \\ Bribe_{P1}(\boldsymbol{IncR}^*, S^*) + R_1^*/3. \end{aligned}$$

Since bribes are non-negative and we assumed $R_1^* > 0$, the difference is strictly positive. The validator's revenue from any other choice remains unchanged. Thus, the validator's unique best response to the deviation bribe functions $\boldsymbol{Bribe}'$ and the random string S^* is to select the report vector $\boldsymbol{IncR}'$.

We have already shown that the deviation makes $\boldsymbol{IncR}'$ the unique best response with random string S^*. We should then show that this deviation strictly increases publisher P_2's

expected revenue. However, before that, we need to ensure that the deviation does not alter the validator's best response for any other random string $S' \neq S^*$. With random string S', the validator may have multiple best responses with identical revenue. The validator possibly selects different best responses related to S' with the original bribe functions ***Bribe*** and with deviating bribe functions ***Bribe'***, even though these two bribe functions are identical for the random string S'. This could affect publisher P_2's revenue with the deviation.

To resolve this, we can let P_2 slightly increase the bribe for the original best response with S' so that it becomes uniquely optimal. As a result, for all random strings other than S^*, the validator's best response to the deviated bribe functions ***Bribe'*** remains unchanged from the original best response with ***Bribe***. Denote by $\mathcal{S}$ the set of all possible random strings and by $\textbf{\textit{IncR}}^*_{S'}$ the original best response with random string S'. We modify the deviation bribe function $\textit{Bribe}'_{P2}$ by distributing $R_1^*/3$ evenly across all possible random string $S' \in \mathcal{S}, S' \neq S^*$. Then, with each random string $S' \neq S^*$, the bribe increases by a small amount of $R_1^*/(3\cdot(|\mathcal{S}|-1))$ only for the original best response $\textbf{\textit{IncR}}^*_{S'}$:

$$\textit{Bribe}'_{P2}(\textbf{\textit{IncR}}, S' \neq S^*) = \begin{cases} \textit{Bribe}_{P2}(\textbf{\textit{IncR}}, S') + \frac{R_1^*}{(3\cdot(|\mathcal{S}|-1))} & \text{if } \textbf{\textit{IncR}} = \textbf{\textit{IncR}}^*_{S'}; \\ \textit{Bribe}_{P2}(\textbf{\textit{IncR}}, S') & \text{otherwise.} \end{cases} \tag{8}$$

Then the original best response $\textbf{\textit{IncR}}^*_{S'}$ becomes the unique best response to the modified deviation bribe function $\textit{Bribe}'_{P2}$ and the random string S'.

(ii) **Publisher P_2's Revenue under Deviation**.

We first calculate publisher P_2's revenue with random string S^* in the deviation case and then calculate her expected revenue over all possible random strings.

Recall that, with the random string S^*, the difference between the validator's original best response $\textbf{\textit{IncR}}^*$ and the new best response $\textbf{\textit{IncR}}'$ is that $\textbf{\textit{IncR}}'$ replaces P_1's report $\textit{Rpt}_{P1}$ in $\textbf{\textit{IncR}}^*$ with P_2's report $\textit{Rpt}_{P2}$. Therefore, due to the symmetric reward property, the reward to publisher P_2 from $\textbf{\textit{IncR}}'$ is the same as the reward to publisher P_1 from $\textbf{\textit{IncR}}^*$:

$$R(\textbf{\textit{IncR}}', \{\textit{Rpt}_{P2}\}, S^*) = R(\textbf{\textit{IncR}}^*, \{\textit{Rpt}_{P1}\}, S^*) \ . \tag{9}$$

The revenue R_1^* of publisher P_1 with the original best response $\textbf{\textit{IncR}}^*$ is the total reward from her included reports minus her bribe paid to the validator:

$$\begin{aligned} R_1^* &= R_{P1}(\{\textit{Rpt}_{P1}\}, \textit{Bribe}_{P1}, \textbf{\textit{IncR}}^*, S^*) \\ &= R(\textbf{\textit{IncR}}^*, \{\textit{Rpt}_{P1}\}, S^*) - \textit{Bribe}_{P1}(\textbf{\textit{IncR}}^*, S^*) \ . \end{aligned} \tag{10}$$

With random string S^* and bribe function $\textit{Bribe}'_{P2}$ the revenue of publisher P_2 due to the new best response $\textbf{\textit{IncR}}'$ of the validator is:

$$\begin{aligned} & R_{P2}(\{\textit{Rpt}_{P2}\}, \textit{Bribe}'_{P2}, \textbf{\textit{IncR}}', S^*) \\ \overset{Eq.1}{=} & R(\textbf{\textit{IncR}}', \{\textit{Rpt}_{P2}\}, S^*) - \textit{Bribe}'_{P2}(\textbf{\textit{IncR}}', S^*) \\ \overset{Eq.5}{=} & R(\textbf{\textit{IncR}}', \{\textit{Rpt}_{P2}\}, S^*) - \\ & (\textit{Bribe}_{P2}(\textbf{\textit{IncR}}^*, S^*) + \textit{Bribe}_{P1}(\textbf{\textit{IncR}}^*, S^*) + R_1^*/3) \\ \overset{Eq.9}{=} & (R(\textbf{\textit{IncR}}^*, \{\textit{Rpt}_{P1}\}, S^*) - \textit{Bribe}_{P1}(\textbf{\textit{IncR}}^*, S^*)) - \\ & (\textit{Bribe}_{P2}(\textbf{\textit{IncR}}^*, S^*) + R_1^*/3) \\ = & R_1^* - \textit{Bribe}_{P2}(\textbf{\textit{IncR}}^*, S^*) - R_1^*/3 \\ \overset{Eq.3}{=} & + (R(\textbf{\textit{IncR}}^*, \{\textit{Rpt}_{P2}\}, S^*) - \textit{Bribe}_{P2}(\textbf{\textit{IncR}}^*, S^*)) + \\ & \frac{2}{3}R_1^* \\ \overset{Eq.1}{=} & \frac{2}{3}R_1^* + R_{P2}(\{\textit{Rpt}_{P2}\}, \textit{Bribe}_{P2}, \textbf{\textit{IncR}}^*, S^*) \ . \end{aligned} \tag{11}$$

With a random string $S' \neq S^*$, publisher P_2's revenue from the deviation bribe function $\textit{Bribe}'_{P2}$ is a bit lower than her revenue from the original bribe function $\textit{Bribe}_{P2}$ due to the additional bribe (Equation 8). Recall that the best responses of the validator $\textbf{\textit{IncR}}^*_{S'}$ in these two cases are identical. We have

$$\begin{aligned} & R_{P2}(\{\textit{Rpt}_{P2}\}, \textit{Bribe}'_{P2}, \textbf{\textit{IncR}}^*_{S'}, S') = \\ & R_{P2}(\{\textit{Rpt}_{P2}\}, \textit{Bribe}_{P2}, \textbf{\textit{IncR}}^*_{S'}, S') - R_1^*/(3\cdot(|\mathcal{S}|-1)). \end{aligned} \tag{12}$$

Denote by $\textit{BR}_V(\cdot)$ the validator's best response. The expected revenue for publisher P_2 with the deviation bribe function $\textit{Bribe}'_{P2}$ is higher than her expected revenue with the original bribe function $\textit{Bribe}_{P2}$, when the validator takes the best responses:

$$\begin{aligned} & E_S[R_{P2}(\{\textit{Rpt}_{P2}\}, \textit{Bribe}'_{P2}, \textit{BR}_V(\cdot), S)] \\ = & \frac{1}{|\mathcal{S}|}\left[\sum_S R_{P2}(\{\textit{Rpt}_{P2}\}, \textit{Bribe}'_{P2}, \textbf{\textit{IncR}}^*_S, S)\right] \\ = & \frac{1}{|\mathcal{S}|}\left[\begin{smallmatrix} R_{P2}(\{\textit{Rpt}_{P2}\}, \textit{Bribe}'_{P2}, \textbf{\textit{IncR}}', S^*) + \\ \sum_{S'\neq S^*} R_{P2}(\{\textit{Rpt}_{P2}\}, \textit{Bribe}'_{P2}, \textbf{\textit{IncR}}^*_{S'}, S') \end{smallmatrix}\right] \\ \overset{Eq.11}{=} & \frac{1}{|\mathcal{S}|}\left[\begin{smallmatrix} \frac{2}{3}R_1^* + R_{P2}(\{\textit{Rpt}_{P2}\}, \textit{Bribe}_{P2}, \textbf{\textit{IncR}}^*, S^*) + \\ \sum_{S'\neq S^*} R_{P2}(\{\textit{Rpt}_{P2}\}, \textit{Bribe}'_{P2}, \textbf{\textit{IncR}}^*_{S'}, S') \end{smallmatrix}\right] \\ \overset{Eq.12}{=} & \frac{1}{|\mathcal{S}|}\left[\begin{smallmatrix} \frac{2}{3}R_1^* + R_{P2}(\{\textit{Rpt}_{P2}\}, \textit{Bribe}_{P2}, \textbf{\textit{IncR}}^*, S^*) + \\ \sum_{S'\neq S^*}\left(R_{P2}(\{\textit{Rpt}_{P2}\}, \textit{Bribe}_{P2}, \textbf{\textit{IncR}}^*_{S'}, S') - \frac{R_1^*}{3\cdot(|\mathcal{S}|-1)}\right) \end{smallmatrix}\right] \\ = & \frac{1}{|\mathcal{S}|}\left[\sum_S R_{P2}(\{\textit{Rpt}_{P2}\}, \textit{Bribe}_{P2}, \textbf{\textit{IncR}}^*_S, S)\right] + \frac{R_1^*}{3|\mathcal{S}|} \\ = & E_S[R_{P2}(\{\textit{Rpt}_{P2}\}, \textit{Bribe}_{P2}, \textit{BR}_V(\cdot), S)] + \frac{R_1^*}{3|\mathcal{S}|} \ . \end{aligned}$$

The above inequality shows that publisher P_2 can increase her expected revenue by deviating to the new bribe function $\textit{Bribe}'_{P2}$. This contradicts the assumption that the protocol is incentive-compatible.

(iii) **Conclusion**.

In all, we have shown that for any situation where publishers have positive expected revenue, there exists a profitable deviation for some publisher, demonstrating that the protocol is not incentive-compatible. This completes the proof of Theorem 1. □

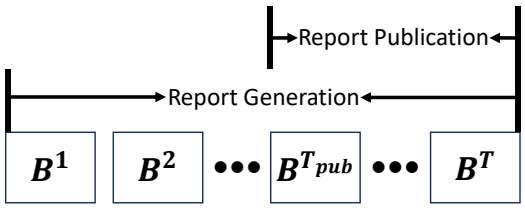

Figure 1: The progress of Prrr in an epoch.

5. Personal Random Rewards for Reporting

To overcome the impossibility of symmetric designs, we create controlled asymmetry among participants (§5.1). Building on this concept, we propose a new reporting protocol, Prrr, which employs a second-price-style reward allocation rule (§5.2) and satisfies our desired goals when all participants adhere to the protocol (§5.3). We analyze the conditions to ensure incentive compatibility (§5.4) and carefully design the random-value function, providing two specific instantiations, to ensure these incentive-compatibility conditions (§5.5).

5.1. Ex-Ante Synthetic Asymmetry (EASA)

As demonstrated in Theorem 1, symmetric reporting protocols cannot achieve our desired goals. To overcome this limitation, we propose *Ex-Ante Synthetic Asymmetry* (EASA), a novel mechanism-design concept that introduces controlled asymmetry among participants.

In traditional mechanism design, asymmetry among participants arises naturally due to differences in their profiles. However, in our reporting setting, all reports are functionally identical, and publishers operate in a permissionless environment. We cannot differentiate between publishers based on the number of reports they generate, as this would compromise fairness and incentivize centralization or identity splitting.

Therefore, EASA assigns a random value to each report using a random-value function *RV*. Given a random string S, the random value of a report *Rpt* is determined as $RV(Rpt, S)$. For a specific random string S, the random values of different reports may vary. However, if the random string S is unbiased, all reports remain functionally identical in expectation.

5.2. The Prrr Protocol

Based on the EASA concept, we design *Prrr*: *personal random rewards for reporting*. The Prrr protocol progresses in steps (§5.2.1), each consisting of three phases corresponding to the reporting problem (§3.4): publication (§5.2.2), inclusion (§5.2.3), and processing (§5.2.4).

5.2.1. Prrr Progress. Assume that an epoch begins at step 1 and spans T steps. The protocol is parameterized by a publication period T_{Pub} (Figure 1).

Algorithm 1 Report Publication by Publisher P_j

1: **Input:** Set of reports the publisher P_j generates: $\boldsymbol{Rpt}_j$
2: $PubR_{Prrr} \leftarrow \{(Rpt, 0) | Rpt \in \boldsymbol{Rpt}_j\}$
3: **Output:** ($PubR_{Prrr}$, $\mathbf{0}$)

- **Report Generation Window:** Publishers may begin generating reports starting from step 1. We can enforce this by requiring reports to include the block hash of B^1, which is only available after the creation of this block.
- **Report Publication Window:** The smart contract will only process published reports starting from step T_{Pub}. Once the protocol reaches the end of the epoch (after step T) or the smart contract receives any report, this window closes, and participants should not take any further actions.

This timeline provides publishers with adequate time to prepare reports before competing for inclusion. Otherwise, if one publisher can generate reports faster than others, they could publish their reports first and secure the reward, leading to centralization.

During the report publication window, in each step, the protocol executes the report publication, inclusion, and reward distribution phases sequentially.

Additionally, in the PoS-pattern report (§3.1), Prrr does not allow publishers to increase their stakes to raise the number of reports they can generate during the report publication window.

5.2.2. Report Publication. Prrr requires publishers to publish all their generated reports without bids or bribes (Algorithm 1). Specifically, for any publisher P_j, denote by $\boldsymbol{Rpt}_j$ the set of reports she has generated. Publisher P_j should publish a set of transactions $PubR_{Prrr}$, where each transaction contains one report from $\boldsymbol{Rpt}_j$ and a zero transaction fee bid. Additionally, she should publish a bribe function that assigns zero to all possible outcomes, denoted by $\mathbf{0}$. Notably, the publisher does not need to be aware of other publishers to execute the above protocol.

5.2.3. Report Inclusion. In any step k during the report publication window, following the report publication phase, Prrr requires validator V^k to generate the random string S^k, and decide which reports to include in block B^k.

We do not want publishers to know the random string S^k when publishing their reports, as this could lead to their strategic publication of reports, deviating from the protocol to get higher revenue. Therefore, Prrr requires each validator V^k to use Verifiable Random Functions (VRFs) [57] to generate the private random string S^k for each block B^k. Anyone can verify its integrity after publication.

The VRF consists of two functions [57]:

- Generation function $GEN_{VRF}(Q^k, SK^k) \rightarrow S^k$: Validator V^k uses their private key SK^k and the public beacon Q^k to generate a private random string S^k.

Algorithm 2 Report Inclusion by Validator V^k

Input: Transaction set from all publishers $\cup_{j=1}^{n} \boldsymbol{PubR}_j$, validator's private key SK^k, block index k, external random source Q^k, and minimum value threshold $r_{\min}$.
function SORT($\cup_{j=1}^{n} \boldsymbol{PubR}_j$, S^k)
    Sort (Rpt, b) in descending order of $RV(Rpt, S^k)$
    Break ties by ascending order of $H(Rpt)$
    Return: Sorted list *SortPubR*
end function
Generate the random string $S^k \leftarrow GEN_{VRF}(Q^k, SK^k)$
$SortPubR \leftarrow$ SORT($\cup_{j=1}^{n} \boldsymbol{PubR}_j, S^k$)
$\boldsymbol{PrInc}^k \leftarrow ()$
if $SortPubR \neq \emptyset$ **then**
    $\boldsymbol{PrInc}^k.append(SortPubR[1])$
end if
if $|SortPubR| > 1$ **then**
    $(Rpt, b) \leftarrow SortPubR[2]$
    if $RV(Rpt, S^k) > r_{\min}$ **then**
        $\boldsymbol{PrInc}^k.append((Rpt, b))$
    end if
end if
Output: $B^k \leftarrow (k, Q^k, S^k, \boldsymbol{PrInc}^k)$

This string is used to calculate the personal rewards for the reports in block B^k and should only be revealed when the block is published.

- Verification function $VER_{VRF}(Q^k, S^k, PK^k) \rightarrow \{\text{true}, \text{false}\}$: After publication, anyone can use the validator's public key PK^k to verify that the private string S^k was correctly generated from the public beacon Q^k.

Prrr adopts a second-price-style report inclusion rule, requiring the validator to include two reports in the block: The first report is the one with the highest random value, designated as the winning report; The second report is the one with the second-highest random value, which we will use to implement the reward allocation mechanism in the next phase. To further enhance efficiency, Prrr requires the validator to include the second-place report only when its value is strictly higher than the minimum possible *RV* value $r_{\min}$, as the smart contract can infer this value during reward allocation without explicit inclusion.

In this phase, validator V^k first generates the secret random string S^k using the VRF generation function. Next, she collects all published reports from publishers and computes their random values using a random-value function *RV*. We denote by $PubR_j$ the set of transactions published by publisher j. Then $\cup_{j=1}^{n} PubR_j$ is the set of all published transactions from all publishers. Finally, she puts the two reports with the highest random values into the block according to the second-price-style rule. The detailed procedure is in Algorithm 2.

Note that the validator does not need to know publishers and only needs to act according to the transactions she receives.

5.2.4. Report Processing. In any step k during the report publication window, after the report inclusion phase, Prrr requires the smart contract to process transactions in block $B^k = (k, Q^k, S^k, \boldsymbol{IncR}^k)$ (Algorithm 3) to allocate rewards.

Our goal in this phase is to allocate rewards to publishers and the validator to incentivize them to follow the protocol. Prrr adopts a second-price-style reward allocation mechanism to achieve this. For intuition, consider first an alternative mechanism where all report values are public, and the report included on chain receives a reward equal to its value. The highest-value report creator must bribe the validator to include it with an amount equal to the second-highest value. Otherwise, the publisher of the second-highest-value report would out-bribe her, receiving the second-highest value but paying less than it, thus earning a positive surplus.

Based on this intuition, Prrr requires the validator to place the two highest-value reports and directly allocates the second-highest value to the validator as her reward, and the winner receives the surplus, i.e., the difference between the highest and second-highest values.

First, the smart contract verifies the integrity of the random string S^k using the VRF verification function. If the verification fails or the vector of transactions $\boldsymbol{IncR}^k$ is empty, the smart contract takes no further action. Otherwise, the smart contract proceeds to allocate rewards to publishers and the validator using a second-price-style reward allocation mechanism. The validator should include the highest and second-highest value reports (if it does not have the minimum value), but in practice this mechanism deals with three scenarios separately.

The first scenario is the *standard case*, which occurs when a block contains exactly two reports and the first, Rpt_1, has a value higher than or equal to the second, Rpt_2. In addition, the second report's value must exceed the minimum value threshold, $r_{\min}$. The publisher of the winning Rpt_1 receives the difference between its value and the second-highest value. The publisher of the second report Rpt_2 does not receive a reward. The validator's reward is the value of the second report. Table 1 provides an example in Case 1: A correctly ordered block with two reports results in the validator earning $R = 8$ (the second report's value), while the winning publisher earns the surplus, $10 - 8 = 2$.

The second scenario is the *succinct case*, which occurs when a block contains only one report. In this case, the publisher of the report receives the difference (maybe 0) between its value and the minimum value, $r_{\min}$. The validator's reward is the minimum value, $r_{\min}$. Table 1 illustrates this in Case 2: A block with a single report results in the validator earning $R = 2$ (the minimum value), while the publisher earns $10 - 2 = 8$.

In all other situations, the mechanism enters a *deviation case*. This occurs if the block contains more than two reports (see Table 1, Case 3), or if it contains two reports but the second report's value is not lower than the first (see Case 4), or if the second report's value equals the minimum value $r_{\min}$ (see Case 5). In these cases, the publisher of the first report receives the difference between its value and the

Algorithm 3 Report Processing by the Smart Contract

Input: Block $B^k = (k, Q^k, S^k, \textbf{\textit{IncR}}^k)$, validator's public key PK^k
Output: Publisher rewards $\mathbf{R}_{\mathcal{P}}$, Validator reward $R_{\mathcal{V}}$
Initialize $\mathbf{R}_{\mathcal{P}} \leftarrow \mathbf{0}$, $R_{\mathcal{V}} \leftarrow 0$
if $\textbf{\textit{IncR}}^k = ()$ or $\mathit{VER}_{\mathit{VRF}}(Q^k, S^k, PK^k) =$ false **then return** $\mathbf{R}_{\mathcal{P}}, R_{\mathcal{V}}$
end if
Note that $\textbf{\textit{IncR}}^k = ((\mathit{Rpt}_1, \cdot), (\mathit{Rpt}_2, \cdot), \cdots)$
Let P_{j*} be the publisher owns Rpt_1
$r_1 \leftarrow \mathit{RV}(\mathit{Rpt}_1, S^k)$
if $|\textbf{\textit{IncR}}^k| = 1$ **then**
    $R_{\mathcal{V}} \leftarrow r_{\min}$
    $\mathbf{R}_{\mathcal{P}}[j^*] \leftarrow r_1 - r_{\min}$
else if $|\textbf{\textit{IncR}}^k| = 2$ **then**
    $r_2 \leftarrow \mathit{RV}(\mathit{Rpt}_2, S^k)$
    if $r_1 \geq r_2$ **and** $r_2 > r_{\min}$ **then**
        $R_{\mathcal{V}} \leftarrow r_2$
        $\mathbf{R}_{\mathcal{P}}[j^*] \leftarrow r_1 - r_2$
    else
        $\mathbf{R}_{\mathcal{P}}[j^*] \leftarrow r_1 - r_{\min}$
    end if
else
    $\mathbf{R}_{\mathcal{P}}[j^*] \leftarrow r_1 - r_{\min}$
end if
return $\mathbf{R}_{\mathcal{P}}, R_{\mathcal{V}}$

	Block Contents ($r_{\min} = 2$)			Reward for Validator	Reward for *Rpt* A	Reward for *Rpt* B	Reward for *Rpt* C
Case 1	*Rpt* A $R = 10$	*Rpt* B $R = 8$		8	$10 - 8 = 2$	0	0
Case 2	*Rpt* A $R = 10$			2	$10 - 2 = 8$	0	0
Case 3	*Rpt* A $R = 10$	*Rpt* B $R = 8$	*Rpt* C $R = 2$	0	$10 - 2 = 8$	0	0
Case 4	*Rpt* B $R = 8$	*Rpt* A $R = 10$		0	0	$8 - 2 = 6$	0
Case 5	*Rpt* A $R = 10$	*Rpt* C $R = 2$		0	$10 - 2 = 8$	0	0

TABLE 1: Revenue calculations in different scenarios.

minimum value $r_{\min}$, while the validator does not receive any reward.

Once the reward allocation is complete, the reporting is finished and the report publication window ends.

Note. In practical systems, three additional considerations arise. First, some blockchain protocols require a minimum fee for each transaction. Second, smart contracts process transactions sequentially within a block, rather than having a global view of the entire block. And third, publishers can specify a validity duration window for each transaction. These considerations do not affect our analysis (§A).

5.3. Correctness

Prrr solves the reporting problem assuming all participants follow the protocol.

Prrr achieves efficiency as it processes at most two reports per epoch ($C = 2$). Prrr also achieves progress because both publishers and validators act promptly to publish and include reports. Furthermore, Prrr achieves **fairness** through randomness: while individual outcomes are asymmetric, every report has an equal probability of winning.

Decentralization requires publishers to have positive expected revenue (ignoring the cost of generating the reports). Prrr requires publishers not to bribe so that revenue is exactly the reward from the smart contract. We denote by $\max^{(2)}$ the second-highest value in a set, by $\mathit{RAllPub}(N)$ the expected total reward for all publishers when there are N reports in total, and by E_S the expectation over the randomness S. According to our design, given any set of N reports $\{\mathit{Rpt}_1, \mathit{Rpt}_2, \ldots, \mathit{Rpt}_N\}$, $\mathit{RAllPub}(N)$ is the expected difference between the highest random value and the second-highest random value among N reports:

$$\mathit{RAllPub}(N) = E_S\left[\max_{1\leq i\leq N} \mathit{RV}(\mathit{Rpt}_i, S) - \max^{(2)}_{1\leq i\leq N} \mathit{RV}(\mathit{Rpt}_i, S)\right] \quad (13)$$

If the random-value function is not constant, this expected difference is strictly positive. By adhering to the protocol, the expected reward for each report is $\frac{1}{N}\mathit{RAllPub}(N)$, as the random value is ex-ante symmetric for all reports. Thus, the expected revenue for a publisher generating N_j reports is $\frac{N_j}{N}\mathit{RAllPub}(N)$. As long as $\mathit{RAllPub}(N) > 0$, every publisher obtains a positive expected revenue by following the protocol, thereby ensuring decentralization.

5.4. Conditions for Incentive Compatibility

Prrr's correctness depends on the assumption that all participants follow the protocol. Here, we analyze how to ensure that following the protocol maximizes each participant's expected revenue, i.e., the protocol is incentive compatible. Although our reward allocation mechanism disincentivizes bribes that alter the order of reports (§5.2.4), we find that we still need two conditions for the random-value function to ensure incentive compatibility: (1) Reward Monotonicity to prevent report withholding; and (2) Skipping Resistance to prevent bribery for skipping steps.

Reward Monotonicity. One challenge for incentive compatibility is that publishers can withhold reports.

The private randomness (§5.2.3) ensures that publishers cannot withhold reports based on their random values. Otherwise, a publisher could only publish her highest-value report, reducing the second-highest value if her report is the highest among all published reports. Therefore, publishers with more reports could disproportionately increase their expected reward per report. This privacy is essential to fairness and incentive compatibility.

However, withholdings can still happen if the expected reward for a publisher decreases when she publishes more reports. If the total reward for all publishers is non-decreasing with the number of reports, then for any publisher, publishing more reports increases both total reward

and her share of the total reward, thus increasing her expected reward. In addition, in the worst case, where there is only one publisher, her expected reward is also non-decreasing with the number of reports she publishes. We call this property *reward monotonicity*:

Property 1 (Reward Monotonicity)**.** *For any two numbers of reports $N_1 < N_2$ up to the maximum possible number of published reports $N_{\max}$, the expected total publisher reward of the larger set must be no less than that of the smaller set:*

$$\forall N_1 < N_2 \leq N_{\max} : \textit{RAllPub}(N_1) \leq \textit{RAllPub}(N_2)$$

Skipping Resistance. Another challenge for incentive compatibility is that publishers might bribe validators to skip certain steps. This allows them to take the chance with new private randomness in the next step, potentially increasing their expected reward. Although publishers cannot predict the random string in advance, they can offer bribes for any potential realization of the random string.

Therefore, if they are not satisfied with the current random string, they may bribe the validator to skip the current step and try another random string in the subsequent step. This is especially true if the validator's reward for honest inclusion is low, while the publisher's potential gain from a new random draw is high. To prevent this, the cost of a successful bribe must exceed the expected gain. The minimum a rational validator would accept as a bribe is their reward from honest inclusion, which is at least $r_{\min}$. We consider the worst-case scenario where all publishers collaborate to bribe the validator. To make such a bribe unprofitable, we require the total expected reward for all publishers to be strictly less than $r_{\min}$. We call this property *skipping resistance*:

Property 2 (Skipping Resistance)**.** *For any number of reports N up to the maximum possible number of published reports $N_{\max}$, the expected total publisher reward is strictly less than the minimum possible validator reward, $r_{\min}$:*

$$\forall 1 \leq N \leq N_{\max} : \textit{RAllPub}(N) < r_{\min}$$

5.5. Random-Value Function Instantiation

We now identify two random-value functions that satisfy Reward Monotonicity (Property 1) and Skipping Resistance (Property 2).

Logarithmic Value Function. Recall that $r_{\min}$ represents the minimum possible output of the random-value function $\textit{RV}_{\textit{log}}(\textit{Rpt}, S)$. Assume that the random oracle $H(\cdot)$ produces values uniformly distributed in the interval $[0, 1)$, and let $\lambda > 0$ be a parameter that controls the distribution of rewards. Given a report *Rpt* and a random string S, we define the logarithmic random-value function as:

$$\textit{RV}_{\textit{log}}(\textit{Rpt}, S) = r_{\min} - \frac{1}{\lambda} \ln(1 - H(\textit{Rpt}||S)) \ .$$

Polarized Value Function. The polarized random-value function has only two possible outcomes: $r_{\min}$ and $r_{\max}$, where $r_{\max} > r_{\min}$ and the probability of receiving $r_{\max}$ is small, denoted by p. Given the random oracle H, a report *Rpt* and a random string S, we define the polarized random-value function $\textit{RV}_{\textit{polarized}}(\textit{Rpt}, S)$ as:

$$\textit{RV}_{\textit{polarized}}(\textit{Rpt}, S) = \begin{cases} r_{\max} & \text{if } H(\textit{Rpt}||S) \leq p; \\ r_{\min} & \text{otherwise.} \end{cases}$$

Properties and Comparison. We prove that both the logarithmic value function and the polarized value function satisfy Reward Monotonicity (Property 1) and Skipping Resistance (Property 2) when their parameters are appropriately chosen. For the polarized value function, parameter selection requires an upper bound $N_{\max}$ on the total number of reports generated by all publishers, whereas the logarithmic value function can be parameterized independently of system scale. Additionally, the expected total reward paid by the smart contract is upper bounded by the constant maximal reward $r_{\max}$ in the polarized value function. Given the number of published reports N, the expected total reward grows as $O(\log N)$ for the logarithmic value function. Further details and analysis are in Appendix B.

6. Game

Our model and protocol give rise to a sequential game (§6.1) played by publishers and validators. We formalize the utilities of participants (§6.2).

6.1. Game Progress and Actions

Given a specified random-value function *RV* for the Prrr protocol, we denote the game for the protocol by $\mathcal{G}(\textit{RV})$ or simply $\mathcal{G}$ when the context is clear. The game is sequential and proceeds in discrete steps where each step corresponds to a model step (§3). The game starts when the report publication phase starts at step T_{Pub} and ends at step T when the epoch ends (§5.2.1).

We assume that publishers cannot generate additional reports since the game starts. For Proof-of-Work reports, this is because the generation of reports is computationally intensive and time-consuming, and we can set the publication window to be short. For Proof-of-Stake reports, generating a report is computationally trivial, but we limit the number of reports by the publisher's stake, which cannot be increased after the game starts (§5.2.1).

To simplify notation, we count the steps of the game from 1 to $m = T - T_{Pub} + 1$ (the length of the publication window) and denote the validator of step $k \in \{1, \ldots, m\}$ by V^k. In each step, publishers publish their reports, and then a validator includes the reports in a block.

We assume all validators follow the protocol to generate the correct random string. Recall that random string generated by validator V^k is S^k. An incorrect S^k would result in the reports being ignored (§5.2.4). Rather than generating

an incorrect S^k, validator V^k could include an empty set of reports, achieving the same outcome.

As discussed in Section 4, we omit bids from the game since bribes can subsume them. We abstract away transactions and focus solely on reports.

We assume that publishers publish their reports in each step without loss of generality since it is equivalent to their actual operations (§4).

6.1.1. Publisher Actions. At the beginning of each step k, publisher P_j publishes a set of reports PubR^k_{Pj} which is a subset of her generated reports $\boldsymbol{\mathit{Rpt}}_{Pj}$, and a bribe function Bribe^k_{Pj}.

6.1.2. Validator Actions. After the publishers act in step k, validator V^k's action, denoted by $\boldsymbol{\mathit{IncR}}^k$, is to select and publish an ordered vector of reports to include in the block. The validator can only choose reports from the set of all reports from publishers in that step (i.e., $\cup_{1\le j\le n}\mathit{PubR}^k_{Pj}$).

Reward allocation in Prrr depends on the validator's inclusion action $\boldsymbol{\mathit{IncR}}^k = (\mathit{IncR}^k_1, \mathit{IncR}^k_2, \ldots, \mathit{IncR}^k_l)$, which yields three possible cases. To simplify subsequent analysis, rather than using the original inclusion action, we reformulate the validator's action. Our goal is to let the reward of the first report become the difference between the values of the first and second reports in all cases. We introduce a dummy report which always has minimum value, denoted by $\mathit{Rpt}_{\text{dummy}}$, to achieve this unified representation:

- The validator includes exactly two reports $(\mathit{IncR}^k_1, \mathit{IncR}^k_2)$, and the value of IncR^k_1 is no less than that of IncR^k_2, and the value of IncR^k_2 is higher than the minimum value. This case already matches our desired structure and we keep it unchanged.
- If the action is an empty vector, we just keep it unchanged.
- In all other cases, including (1) the validator includes more than two reports; (2) the validator includes only one report; and (3) the validator includes exactly two reports $(\mathit{IncR}^k_1, \mathit{IncR}^k_2, \ldots)$ but the value of IncR^k_1 is less than that of IncR^k_2 or the value of IncR^k_2 is the minimum value. We insert the dummy report after the first report: $\boldsymbol{\mathit{IncR}}^k = (\mathit{IncR}^k_1, \mathit{Rpt}_{\text{dummy}}, \mathit{IncR}^k_2, \ldots, \mathit{IncR}^k_l)$.

Note that, after this reformulation, if we insert the second position, all subsequent reports are shifted one position to the right. For consistency, we mark $\mathit{Rpt}_{\text{dummy}}$ as IncR^k_2 and mark IncR^k_i where $i \ge 2$ as IncR^k_{i+1}. After the reformulation, if the action $\boldsymbol{\mathit{IncR}}^k$ contains exactly two reports, the validator's reward is the value of the second report in the first two cases; otherwise, the validator's reward is zero. We assume that publishers provide bribes based on the reformulated inclusion action.

6.1.3. Actions Following Prrr Protocol. We now define the actions that follow the Prrr protocol (§5).

For a validator V^k, following the Prrr protocol means selecting the report with the highest random value as the first included report. If the second-highest value among the published reports exceeds the minimum value, the validator includes that report as the second included report; otherwise, the dummy report $\mathit{Rpt}_{\text{dummy}}$ is included as the second report. If multiple reports share the same value, Prrr adopts a tie-breaking rule based on the hash to get a total order (Algorithm 3). We denote this action by $\boldsymbol{\mathit{IncR}}^k = \boldsymbol{\mathit{PrInc}}$.

For a publisher P_j, following the Prrr protocol means she publishes all her reports (since we abstract away transactions), i.e., $\mathit{PubR}^k_{Pj} = \boldsymbol{\mathit{Rpt}}_{Pj}$, and that they do not offer any bribes, i.e., $\mathit{Bribe}^k_{Pj} = \mathbf{0}$, as specified in Algorithm 1.

6.2. Utilities

We now formalize the utilities of validators and publishers in the game.

6.2.1. Validator Utility. In step k, for a given action $\boldsymbol{\mathit{IncR}}^k = (\mathit{IncR}^k_1, \mathit{IncR}^k_2, \ldots)$ of validator V^k, the bribe functions $\boldsymbol{\mathit{Bribe}}^k = (\mathit{Bribe}^k_{P1}, \cdots, \mathit{Bribe}^k_{Pn})$ from all publishers, and the random string S^k, the revenue of validator V^k is the reward from the smart contract plus the sum of all bribes. Since a validator only acts in a single step, that revenue is her utility. We denote by $\mathcal{I}$ the indicator function, that is, $\mathcal{I}(\mathit{True}) = 1$ and $\mathcal{I}(\mathit{False}) = 0$. The utility of validator V^k is thus:

$$U^k_V(\boldsymbol{\mathit{IncR}}^k = (\mathit{IncR}^k_1, \mathit{IncR}^k_2, \ldots), \boldsymbol{\mathit{Bribe}}^k, S^k) = \mathcal{I}(|\boldsymbol{\mathit{IncR}}^k| = 2) \cdot \mathit{RV}(\mathit{IncR}^k_2, S^k) + \sum_{j=1}^{n} \mathit{Bribe}^k_{Pj}(\boldsymbol{\mathit{IncR}}^k, S^k). \quad (14)$$

6.2.2. Publisher Utility. The report generation cost of each publisher is constant and independent of their actions in the game, so we omit it from the utility function.

The utility of a publisher is the sum of her revenue across all steps in the epoch. We first focus on the revenue in a single step. Given publisher P_j's action $(\mathit{PubR}^k_{Pj}, \mathit{Bribe}^k_{Pj})$, the random string S^k, and the subsequent action of the validator $\boldsymbol{\mathit{IncR}}^k = (\mathit{IncR}^k_1, \mathit{IncR}^k_2, \ldots)$, the revenue of publisher P_j in step k is the revenue from reports (only when included as the first report) minus bribes paid:

$$R^k_{Pj}(\mathit{PubR}^k_{Pj}, \mathit{Bribe}^k_{Pj}, \boldsymbol{\mathit{IncR}}^k, S^k) = \mathcal{I}(\mathit{IncR}^k_1 \in \mathit{PubR}^k_{Pj}) \cdot (\mathit{RV}(\mathit{IncR}^k_1, S^k) - \mathit{RV}(\mathit{IncR}^k_2, S^k)) - \mathit{Bribe}^k_{Pj}(\boldsymbol{\mathit{IncR}}^k, S^k)\ . \quad (15)$$

The utility of publisher P_j over the entire epoch is the expected sum of her revenues R^k_{Pj} across all steps and random strings.

7. Incentive Compatibility of Prrr

We first analyze the best response of participants in the game (§7.1). We prove that all participants following the protocol is an equilibrium (§7.2) robust against collusion, Sybil attacks and deviations that affect the revenue of other participants (§7.3).

7.1. Best Response

We use a standard solution concept for our sequential game, the Subgame Perfect Nash Equilibrium (SPNE). Denote the subgame from step k to the last step m by $\mathcal{G}^k$. In the SPNE, in every step k, each participant selects a best response that maximizes their utility in the subgame $\mathcal{G}^k$, assuming all participants will act optimally in subsequent subgames $\mathcal{G}^{k+1}$.

7.1.1. Validator's Best Response. Unlike publishers, a validator only acts in a single step, and her utility only depends on the actions of publishers and her own action in that step (Equation 14). Therefore, her best response is independent of previous and future steps. For a validator V^k in step k, her best response $BR_V^k(\boldsymbol{PubR}^k, \boldsymbol{Bribe}^k, S^k)$ is simply to maximize her utility (Equation 14) based on the actions of all publishers in that step $\boldsymbol{PubR}^k = (PubR_{P1}^k, \ldots, PubR_{Pn}^k)$ and $\boldsymbol{Bribe}^k = (Bribe_{P1}^k, \ldots, Bribe_{Pn}^k)$ and the random string S^k. So for all validator actions $\boldsymbol{IncR}^k$:

$$U_V^k(BR_V^k(\boldsymbol{PubR}^k, \boldsymbol{Bribe}^k, S^k), \boldsymbol{Bribe}^k, S^k) \geq U_V^k(\boldsymbol{IncR}^k, \boldsymbol{Bribe}^k, S^k).$$

We denote $BR_V^k(\boldsymbol{PubR}^k, \boldsymbol{Bribe}^k, S^k)$ by $BR_V^k(\cdot)$ when the parameters are clear from the context.

7.1.2. Publisher Best Response. For a publisher P_j, the best response in step k depends on the actions and random strings in all subsequent steps since she acts and earns revenue in all steps of the game. Therefore, we use backward induction to analyze the best response.

If no validator has included any report before step k, the game proceeds in step k. The revenue of validator V^k in step k (Equation 14) and the revenue of publisher P_j in step k (Equation 15) depend only on the actions in step k and the random string S^k. Denote by $R_{Pj}(\mathcal{G}^k)$ the expected revenue of publisher P_j in subgame $\mathcal{G}^k$ when all participants follow the best response strategies.

Given the revenue of the subgame starting from next step $R_{Pj}(\mathcal{G}^{k+1})$, the action of publisher P_j in this step $(PubR_{Pj}^k, Bribe_{Pj}^k)$, the actions of other publishers in this step $\boldsymbol{PubR}_{-Pj}^k$ and $\boldsymbol{Bribe}_{-Pj}^k$, and the best response of the validator $BR_V^k(\cdot)$ to these actions, the utility of publisher P_j in the subgame $\mathcal{G}^k$ is the expected revenue in step k plus the expected revenue in the subsequent subgame if validator V^k does not include any report (thus the game continues):

$$U_{Pj}^{k\sim m}(PubR_{Pj}^k, Bribe_{Pj}^k, \boldsymbol{PubR}_{-Pj}^k, \boldsymbol{Bribe}_{-Pj}^k) = E_{S^k}\left[\begin{matrix} R_{Pj}^k(PubR_{Pj}^k, Bribe_{Pj}^k, BR_V^k(\cdot), S^k)+ \\ \mathcal{I}(BR_V^k(\cdot)=())\cdot R_{Pj}(\mathcal{G}^{k+1}) \end{matrix}\right]. \quad (16)$$

Denote by $BR_j^k(\boldsymbol{PubR}_{-Pj}^k, \boldsymbol{Bribe}_{-Pj}^k, R_{Pj}(\mathcal{G}^{k+1}))$ the best response of publisher P_j in step k. We denote it by $BR_j(\cdot)$ when the parameters are clear from the context. For any action $(PubR_{Pj}^k, Bribe_{Pj}^k)$ in step k, the best response strategy has no less utility:

$$U_{Pj}^{k\sim m}(BR_j(\cdot), \boldsymbol{PubR}_{-Pj}^k, \boldsymbol{Bribe}_{-Pj}^k) \geq U_{Pj}^{k\sim m}(PubR_{Pj}^k, Bribe_{Pj}^k, \boldsymbol{PubR}_{-Pj}^k, \boldsymbol{Bribe}_{-Pj}^k). \quad (17)$$

7.2. Prrr Equilibrium

We show that publishers and validators following the Prrr protocol constitutes an SPNE. We prove this using backward induction, starting from the final step of the game and determining the best response for each player at each step, assuming all future players will also act optimally.

Recall that the action following the Prrr protocol for all publishers P_j is to publish all reports and provide a bribe function that always outputs 0, i.e., $(\boldsymbol{Rpt}_{Pj}, \boldsymbol{0})$. And for a validator V^k the desired action $\boldsymbol{PrInc}$ is to include the report with the highest random value as the first report and the report with the second-highest random value as the second report. At this point, we do not rely on the specific random-value function RV but only require it to satisfy Reward Monotonicity (Property 1) and Skipping Resistance (Property 2).

7.2.1. Single-Step Analysis. To analyze the full game, we first analyze the case of a single step to get some preliminary insights.

We first focus on the case where all publishers do not offer any bribes in step k. Then, the revenue of validator V^k (Equation 14) only comes from the reward from the smart contract. The reward is at most the second-highest random value among all published reports, which is exactly the reward she would get by following the Prrr protocol. Therefore, following the Prrr protocol is a best response for the validator.

Observation 1. *In step k, if all publishers do not provide bribes, following the Prrr protocol is a best response for validator V^k.*

For any publisher, according to Equation 15, we have:

Observation 2. *If publisher P_j does not provide any bribe in step k, her revenue in this step is non-negative.*

For any publisher P_j with action $(PubR_{Pj}^k, Bribe_{Pj}^k)$ in step k, if she provides no bribe (i.e., $Bribe_{Pj}^k = \boldsymbol{0}$), her expected revenue is simply the expected reward from the smart contract (see Equation 15). When validator V^k follows the Prrr protocol, each report has an equal probability to win the reward. Thus, the expected revenue for publisher P_j in this step is the proportion of the number of reports

she publishes times the expected total reward $\mathit{RAllPub}(\cdot)$ distributed by the smart contract:

Observation 3. *Given any report set PubR^k_{Pj} published by publisher P_j in step k, if she provides no bribe (i.e., $\mathit{Bribe}^k_{Pj} = \mathbf{0}$) and validator V^k follows the Prrr protocol, the expected revenue of publisher P_j in this step is:*

$$E_{S^k}[R^k_{Pj}(\mathit{PubR}^k_{Pj}, \mathbf{0}, \boldsymbol{\mathit{PrInc}}, S^k)] = \frac{|\mathit{PubR}^k_{Pj}|}{\sum_{1\le i\le n}|\mathit{PubR}^k_{Pi}|} \cdot \mathit{RAllPub}\left(\sum_{1\le i\le n}|\mathit{PubR}^k_{Pi}|\right). \quad (18)$$

If P_j publishes all her reports (i.e., $\mathit{PubR}^k_{Pj} = \mathit{Rpt}_j$), the total reward $\mathit{RAllPub}(\cdot)$ is maximized due to Reward Monotonicity (Property 1). The proportion of her reports $\frac{|\mathit{PubR}^k_{Pj}|}{\sum_{1\le i\le n}|\mathit{PubR}^k_{Pi}|}$ is also maximized. Therefore, when the validator follows the Prrr protocol, any publisher maximizes her expected revenue by publishing all her reports without providing any bribe.

Observation 4. *If the random-value function satisfies Reward Monotonicity (Property 1), then in step k, if all publishers provide no bribe ($\mathit{Bribe}^k_{Pi} = \mathbf{0}$ for all i) and the validator follows the Prrr protocol, any publisher P_j has no less expected revenue by publishing all her reports $\boldsymbol{\mathit{Rpt}}_j$ than by publishing any subset of her reports $\mathit{PubR}^k_{Pj} \subseteq \boldsymbol{\mathit{Rpt}}_j$:*

$$E_{S^k}[R^k_{Pj}(\boldsymbol{\mathit{Rpt}}_j, \mathbf{0}, \boldsymbol{\mathit{PrInc}}, S^k)] \ge E_{S^k}[R^k_{Pj}(\mathit{PubR}^k_{Pj}, \mathbf{0}, \boldsymbol{\mathit{PrInc}}, S^k)] .$$

Now we turn to the case where publisher P_j provides a bribe in step k while all other publishers follow the Prrr protocol. We prove that such bribery does not increase her single-step revenue compared to publishing the same report set without any bribe:

Lemma 1. *In step k with any random string S^k, assume that all publishers other than publisher P_j follow the Prrr protocol in step k. Then, the revenue of publisher P_j with any action $(\mathit{PubR}^k_{Pj}, \mathit{Bribe}^k_{Pj})$ in step k and any best response $\mathit{BR}^k_V(\cdot)$ of validator V^k is no higher than her revenue with the same report set but no bribe, $(\mathit{PubR}^k_{Pj}, \mathbf{0})$, if the validator follows the Prrr protocol:*

$$R^k_{Pj}(\mathit{PubR}^k_{Pj}, \mathit{Bribe}^k_{Pj}, \mathit{BR}_V(\cdot), S^k) \le R^k_{Pj}(\mathit{PubR}^k_{Pj}, \mathbf{0}, \boldsymbol{\mathit{PrInc}}, S^k) \quad (19)$$

Intuitively, we first calculate the upper bound of the combined revenue of publisher P_j and the utility of validator V^k. Next, we determine the lower bound of the utility of validator V^k with her best response. Using these bounds, we derive an upper limit for the revenue of publisher P_j and compare it to her revenue when she offers no bribe.

Proof. When the publisher P_j takes action $(\mathit{PubR}^k_{Pj}, \mathbf{0})$, according to Observation 2, since there is no bribery from P_j, her revenue is non-negative. If taking action $(\mathit{PubR}^k_{Pj}, \mathit{Bribe}^k_{Pj})$ has higher revenue, its revenue must be positive. This is only possible when the validator V^k includes P_j's report as the first report in any of her best response (Equation 14), i.e., $\mathit{Rpt}^k_1 \in \mathit{PubR}^k_{Pj}$.

Since all other publishers follow the Prrr protocol, which does not provide bribe, the revenue of the validator V^k (Equation 14) with action $\boldsymbol{\mathit{IncR}}^k = (\mathit{Rpt}^k_1, \mathit{Rpt}^k_2, \ldots)$ is:

$$U^k_V(\boldsymbol{\mathit{IncR}}^k, \boldsymbol{\mathit{Bribe}}^k, S^k) = \mathcal{I}(|\boldsymbol{\mathit{IncR}}^k| = 2) \cdot \mathit{RV}(\mathit{Rpt}^k_2, S^k) + \mathit{Bribe}^k_{Pj}(\boldsymbol{\mathit{IncR}}^k, S^k).$$

Since $\mathit{Rpt}^k_1 \in \mathit{PubR}^k_{Pj}$, the sum of the revenue of publisher P_j and the utility of validator V^k is:

$$\begin{aligned}
&R^k_{Pj}(\mathit{PubR}^k_{Pj}, \mathit{Bribe}^k_{Pj}, \boldsymbol{\mathit{IncR}}^k, S^k) + \\
&U^k_V(\boldsymbol{\mathit{IncR}}^k, \boldsymbol{\mathit{Bribe}}^k, S^k) \\
=&\mathcal{I}(\mathit{Rpt}^k_1 \in \mathit{PubR}^k_{Pj}) \cdot (\mathit{RV}(\mathit{Rpt}^k_1, S^k) - \mathit{RV}(\mathit{Rpt}^k_2, S^k)) + \\
&\mathcal{I}(|\boldsymbol{\mathit{IncR}}^k| = 2) \cdot \mathit{RV}(\mathit{Rpt}^k_2, S^k) \\
\le&(\mathit{RV}(\mathit{Rpt}^k_1, S^k) - \mathit{RV}(\mathit{Rpt}^k_2, S^k)) + \mathit{RV}(\mathit{Rpt}^k_2, S^k) \\
\le&\mathit{RV}(\mathit{Rpt}^k_1, S^k) \\
\le& \max_{\mathit{Rpt}\in\mathit{PubR}^k_{Pj}} \mathit{RV}(\mathit{Rpt}, S^k) .
\end{aligned}$$

By following the Prrr protocol, the utility of the validator V^k is the second-highest random value among all published reports, i.e., $\max^{(2)}_{\mathit{Rpt}\in\cup^n_{i=1}\mathit{PubR}^k_{Pi}} \mathit{RV}(\mathit{Rpt}, S^k)$, plus the bribe from publisher P_j:

$$U^k_V(\boldsymbol{\mathit{PrInc}}, \boldsymbol{\mathit{Bribe}}^k, S^k) = \max^{(2)}_{\mathit{Rpt}\in\cup^n_{i=1}\mathit{PubR}^k_{Pi}} \mathit{RV}(\mathit{Rpt}, S^k) + \mathit{Bribe}^k_{Pj}(\boldsymbol{\mathit{PrInc}}, S^k).$$

The revenue of the best response of validator V^k is no less than the revenue when following the Prrr protocol. Hence, the revenue of publisher P_j with action $(\mathit{PubR}^k_{Pj}, \mathit{Bribe}^k_{Pj})$ is bounded by:

$$R^k_{Pj}(\mathit{PubR}^k_{Pj}, \mathit{Bribe}^k_{Pj}, \mathit{BR}_V(\cdot), S^k) \le \max_{\mathit{Rpt}\in\mathit{PubR}^k_{Pj}} \mathit{RV}(\mathit{Rpt}, S^k) - \max^{(2)}_{\mathit{Rpt}\in\cup^n_{i=1}\mathit{PubR}^k_{Pi}} \mathit{RV}(\mathit{Rpt}, S^k) - \mathit{Bribe}^k_{Pj}(\boldsymbol{\mathit{IncR}}^k, S^k). \quad (20)$$

We now turn to the case where the publisher P_j adopts the action $(\mathit{PubR}^k_{Pj}, \mathbf{0})$ and validator V^k follows the Prrr protocol. If P_j's published report has the highest random value, i.e., $\max_{\mathit{Rpt}\in\mathit{PubR}^k_{Pj}} \mathit{RV}(\mathit{Rpt}, S^k) = \max_{\mathit{Rpt}\in\cup^n_{i=1}\mathit{PubR}^k_{Pi}} \mathit{RV}(\mathit{Rpt}, S^k)$, her revenue is the difference between the highest and the second-highest random values among all published reports; if not, her revenue is zero:

$$R^k_{Pj}(\mathit{PubR}^k_{Pj}, \mathbf{0}, \boldsymbol{\mathit{PrInc}}, S^k) = \mathcal{I}\left(\max_{\mathit{Rpt}\in\mathit{PubR}^k_{Pj}} \mathit{RV}(\mathit{Rpt}, S^k) = \max_{\mathit{Rpt}\in\cup^n_{i=1}\mathit{PubR}^k_{Pi}} \mathit{RV}(\mathit{Rpt}, S^k)\right) \cdot$$

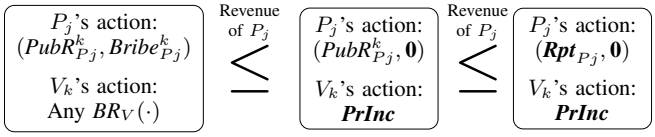

Figure 2: Proof sketch of Theorem 2.

$$\left(\max_{Rpt \in PubR^k_{Pj}} RV(Rpt, S^k) - \max^{(2)}_{Rpt \in \cup_{i=1}^n PubR^k_{Pi}} RV(Rpt, S^k)\right) \tag{21}$$

If P_j's published report has the highest random value, the revenue with action $(PubR^k_{Pj}, \mathbf{0})$ is (Equation 21)

$$\max_{Rpt \in PubR^k_{Pj}} RV(Rpt, S^k) - \max^{(2)}_{Rpt \in \cup_{i=1}^n PubR^k_{Pi}} RV(Rpt, S^k),$$

which is no less than the revenue with action $(PubR^k_{Pj}, Bribe^k_{Pj})$ (Equation 20). Otherwise, if P_j's published report does not have the highest random value, the revenue with action $(PubR^k_{Pj}, \mathbf{0})$ is zero (Equation 21). In this case, the highest random value of reports published by publisher P_j is no higher than the second-highest random value among all published reports, i.e., $\max_{Rpt \in PubR^k_{Pj}} RV(Rpt, S^k) \leq \max^{(2)}_{Rpt \in \cup_{i=1}^n PubR^k_{Pi}} RV(Rpt, S^k)$. Therefore, the revenue with action $(PubR^k_{Pj}, Bribe^k_{Pj})$ (Equation 20) is no higher than zero, which is again no higher than the revenue with action $(PubR^k_{Pj}, \mathbf{0})$ (Equation 21):

$$R^k_{Pj}(PubR^k_{Pj}, Bribe^k_{Pj}, BR_V(\cdot), S^k) \leq R^k_{Pj}(PubR^k_{Pj}, \mathbf{0}, \textbf{PrInc}, S^k). \quad \square$$

7.2.2. Full Game Equilibrium. We now consider the SPNE of the full game $\mathcal{G}$.

Theorem 2. *In the game $\mathcal{G}(RV)$, if the random-value function RV satisfies Reward Monotonicity (Property 1) and Skipping Resistance (Property 2), then all participants following the Prrr protocol constitute a Subgame Perfect Nash Equilibrium (SPNE).*

We prove this theorem by backward induction. By Observation 1, validators have no incentive to deviate from the Prrr protocol when all publishers follow it, as there are no bribes. For any publisher P_j, at each step of the backward induction, we first show that her expected revenue from any deviation $(PubR^k_{Pj}, Bribe^k_{Pj})$ is no higher than from taking $(PubR^k_{Pj}, \mathbf{0})$ (Lemma 1). Next, we show that publishing all reports without bribes $(Rpt_j, \mathbf{0})$ yields at least as much expected revenue as any subset $(PubR^k_{Pj}, \mathbf{0})$ (Observation 4). The proof structure is illustrated in Figure 2.

Proof. By Observation 1, validators have no incentive to deviate from the Prrr protocol when all publishers follow it, since there is no bribery. Thus, it suffices to show that no publisher can profitably deviate from the Prrr protocol. We prove this by backward induction. We first prove that in the subgame $\mathcal{G}^m$, which is just the last step m, no publisher can profitably deviate. Then, we consider the subgame $\mathcal{G}^k$ starting from step k. Assuming that in the subgame $\mathcal{G}^{k+1}$ (starting from the next step), all participants follow the Prrr protocol, we prove that no publisher can profitably deviate in step k as well.

(1) Subgame $\mathcal{G}^m$ starting from the last step m.

In Lemma 1, we established that bribery cannot increase a publisher's revenue when all other publishers follow the Prrr protocol and the validator adopts any best response. It remains to show that publishing fewer reports does not improve a publisher's expected revenue when all other publishers follow the Prrr protocol and the validator follows the protocol as well. This follows directly from Observation 4.

Assume by contradiction that a publisher P_j deviates from the Prrr protocol in step m to any action $(PubR^m_{Pj}, Bribe^m_{Pj})$ while all other publishers follow the Prrr protocol and validator V^m follows any best response. According to Equation 19, the publisher P_j's revenue in this case is not higher than her revenue with action $(PubR^m_{Pj}, \mathbf{0})$, which is the same report set but without bribe, if validator V^m follows the Prrr protocol, for any random string S^m. Therefore, the following holds for the expected revenue over the distribution of random string S^m:

$$E_{S^m}[R^m_{Pj}(PubR^m_{Pj}, Bribe^m_{Pj}, BR_V(\cdot), S^m)] \leq E_{S^m}[R^m_{Pj}(PubR^m_{Pj}, \mathbf{0}, \textbf{PrInc}, S^m)]\ . \tag{22}$$

This is the first inequality in Figure 2.

Now we turn to compare the expected revenue of publisher P_j with action $(PubR^m_{Pj}, \mathbf{0})$ and action following the Prrr protocol, i.e., $(\boldsymbol{Rpt}_{Pj}, \mathbf{0})$, when validator V^m follows the Prrr protocol. According to Observation 4, since the value function satisfies the monotonicity property (Property 1), the expected revenue of publisher P_j with action $(PubR^m_{Pj}, \mathbf{0})$ is no higher than the expected revenue with action $(\boldsymbol{Rpt}_{Pj}, \mathbf{0})$, when validator V^m follows the Prrr protocol:

$$E_{S^m}[R^m_{Pj}(PubR^m_{Pj}, \mathbf{0}, \textbf{PrInc}, S^m)] \leq E_{S^m}[R^m_{Pj}(\boldsymbol{Rpt}_{Pj}, \mathbf{0}, \textbf{PrInc}, S^m)]$$

Combining Equation 22 with the previous analysis, we conclude that for publisher P_j, if all other publishers follow the Prrr protocol, her expected revenue from any action $(PubR^m_{Pj}, Bribe^m_{Pj})$, with validator V^m taking any best response, is no higher than her expected revenue from publishing all reports with zero bribe $(\boldsymbol{Rpt}_{Pj}, \mathbf{0})$ when validator V^m follows the Prrr protocol:

$$E_{S^m}[R^m_{Pj}(PubR^m_{Pj}, Bribe^m_{Pj}, BR_V(\cdot), S^m)] \leq E_{S^m}[R^m_{Pj}(\boldsymbol{Rpt}_{Pj}, \mathbf{0}, \textbf{PrInc}, S^m)]\ .$$

This is the second inequality in Figure 2.

Since there is no subsequent step, the utility of publisher P_j in subgame $\mathcal{G}^m$ is exactly her revenue in step m according to Equation 16:

$$U^m_{Pj}(PubR^m_{Pj}, Bribe^m_{Pj}, \boldsymbol{PubR}^m_{-Pj}, \boldsymbol{Bribe}^m_{-Pj}) =$$

$$E_{S^m}\left[R^m_{Pj}(\textit{PubR}^m_{Pj}, \textit{Bribe}^m_{Pj}, \textit{BR}^m_V(\cdot), S^m)\right].$$

Therefore, for publisher P_j, her utility in subgame $\mathcal{G}^m$ with any action $(\textit{PubR}^m_{Pj}, \textit{Bribe}^m_{Pj})$, considering any best response of the validator V^m, is no higher than her utility with action $(\boldsymbol{Rpt}_{Pj}, \mathbf{0})$ when validator V^m follows the Prrr protocol.

(2) Subgame $\mathcal{G}^k$ starting from any step $1 \le k < m$.

Assume that all participants follow the Prrr protocol in the subgame $\mathcal{G}^{k+1}$ starting from the next step. Unlike the base case, we cannot immediately conclude from Lemma 1 that bribery does not increase the publisher's utility, since the utility in $\mathcal{G}^k$ includes the expected revenue in $\mathcal{G}^{k+1}$ if validator V^k chooses not to include any report. To address this, we show that if a publisher offers a bribe high enough to induce the validator to skip inclusion, her expected revenue does not increase, due to Skipping Resistance (Property 2). Therefore, as in the base case, providing bribes does not increase the expected revenue of the publisher. Similarly, by Observation 4, publishing fewer reports does not increase the expected revenue of a publisher.

Assume that any publisher P_j deviates from the Prrr protocol in step k to any action $(\textit{PubR}^k_{Pj}, \textit{Bribe}^k_{Pj})$. We denote by $R^{k\sim m}_{Pj}(\textit{PubR}^k_{Pj}, \textit{Bribe}^k_{Pj}, \textbf{\textit{IncR}}, S^k, R_{Pj}(\mathcal{G}^{k+1}))$ the revenue of publisher P_j in subgame $\mathcal{G}^k$ with action $(\textit{PubR}^k_{Pj}, \textit{Bribe}^k_{Pj})$ if the validator V^k takes action $\textbf{\textit{IncR}}$ with random string S^k and the expected revenue of publisher P_j in subgame $\mathcal{G}^{k+1}$ is $R_{Pj}(\mathcal{G}^{k+1})$. This revenue is

$$\begin{aligned} R^{k\sim m}_{Pj}(\textit{PubR}^k_{Pj}, \textit{Bribe}^k_{Pj}, \textbf{\textit{IncR}}, S^k, R_{Pj}(\mathcal{G}^{k+1})) = \\ R^k_{Pj}(\textit{PubR}^k_{Pj}, \textit{Bribe}^k_{Pj}, \textbf{\textit{IncR}}, S^k) + \\ \mathcal{I}\left(\textbf{\textit{IncR}} = ()\right) \cdot R_{Pj}(\mathcal{G}^{k+1}) \ . \end{aligned}$$

We first prove that, if all other publishers follow the Prrr protocol, the expected revenue of publisher P_j in subgame $\mathcal{G}^k$ with action $(\textit{PubR}^k_{Pj}, \textit{Bribe}^k_{Pj})$ if the validator V^k follows any best response, is no higher than the expected revenue with action $(\textit{PubR}^k_{Pj}, \mathbf{0})$ if the validator V^k follows the Prrr protocol, for any random string S^k, i.e.,

$$\begin{aligned} R^{k\sim m}_{Pj}(\textit{PubR}^k_{Pj}, \textit{Bribe}^k_{Pj}, \textit{BR}_V(\cdot), S^k, R_{Pj}(\mathcal{G}^{k+1})) \le \\ R^{k\sim m}_{Pj}(\textit{PubR}^k_{Pj}, \mathbf{0}, \textbf{\textit{PrInc}}, S^k, R_{Pj}(\mathcal{G}^{k+1})). \end{aligned}$$

We first analyze the revenue of publisher P_j with action $(\textit{PubR}^k_{Pj}, \mathbf{0})$ if the validator V^k follows the Prrr protocol. Due to Observation 2, since all participants follow the Prrr protocol in subgame $\mathcal{G}^{k+1}$ by the induction hypothesis, the expected revenue of publisher P_j in subgame $\mathcal{G}^{k+1}$ is non-negative, i.e., $R_{Pj}(\mathcal{G}^{k+1}) \ge 0$. Also due to Observation 2, the revenue of publisher P_j in step k without bribe is non-negative, i.e., $R^k_{Pj}(\textit{PubR}^k_{Pj}, \mathbf{0}, \textbf{\textit{PrInc}}, S^k) \ge 0$. Therefore, publisher P_j's revenue in subgame $\mathcal{G}^k$ with action $(\textit{PubR}^k_{Pj}, \mathbf{0})$ is non-negative:

$$R^{k\sim m}_{Pj}(\textit{PubR}^k_{Pj}, \mathbf{0}, \textbf{\textit{PrInc}}, S^k, R_{Pj}(\mathcal{G}^{k+1})) \ge 0 \qquad (23)$$

We now analyze the revenue of publisher P_j with action $(\textit{PubR}^k_{Pj}, \textit{Bribe}^k_{Pj})$ if the validator V^k follows any best response. This includes two possible cases: (i) the best response of validator V^k is to include no report, i.e., $\textit{BR}_V(\cdot) = ()$; or (ii) the best response of validator V^k is to include some reports, i.e., $\textit{BR}_V(\cdot) \ne ()$. We prove that in both cases, the revenue of publisher P_j with action $(\textit{PubR}^k_{Pj}, \textit{Bribe}^k_{Pj})$ is no higher than the revenue with action $(\textit{PubR}^k_{Pj}, \mathbf{0})$ if the validator V^k follows the Prrr protocol.

(i) If the best response of validator V^k is to include no report with random string S^k, the revenue of publisher P_j is her expected revenue in subgame $\mathcal{G}^{k+1}$ minus the bribe she gives to the validator V^k for including no report.

Recall that $R_{Pj}(\mathcal{G}^{k+1})$ is the expected revenue for publisher P_j in the next subgame, which equals her report share times the total expected publisher reward. Therefore, the expected revenue of publisher P_j in subgame $\mathcal{G}^{k+1}$ is no higher than the total expected publisher reward $\textit{RAllPub}(\sum_i |\boldsymbol{Rpt}_i|)$. By Skipping Resistance (Property 2), the total expected publisher reward $\textit{RAllPub}(\cdot)$ is at most $r_{\min}$. Therefore, the expected revenue of publisher P_j in subgame $\mathcal{G}^{k+1}$ is upper bounded by $r_{\min}$:

$$R_{Pj}(\mathcal{G}^{k+1}) \le \textit{RAllPub}(\sum_i |\boldsymbol{Rpt}_i|) \le r_{\min} \ . \qquad (24)$$

By following the Prrr protocol, the validator V^k can get at least $r_{\min}$ from the smart contract. Since including no report is the best response of validator V^k, her revenue with this action must be no less than the revenue by following the Prrr protocol, hence no less than $r_{\min}$. When validator V^k includes no report with random string S^k, she can only get revenue from the bribe from publisher P_j (Equation 14). This is because all other publishers follow the Prrr protocol that does not provide bribe and the protocol does not reward the validator for including no report. Therefore, the bribe given by publisher P_j on the empty inclusion vector is no less than $r_{\min}$, i.e., $\textit{Bribe}^k_{Pj}((), S^k) \ge r_{\min}$.

Recall that the expected revenue of publisher P_j is her expected revenue in subgame $\mathcal{G}^{k+1}$ minus the bribe she gives to the validator V^k. Since for publisher P_j, the bribe is no less than $r_{\min}$ and the expected revenue in subgame $\mathcal{G}^{k+1}$ is no higher than $r_{\min}$ (Equation 24), the expected revenue of publisher P_j is non-positive:

$$\begin{aligned} & R^{k\sim m}_{Pj}(\textit{PubR}^k_{Pj}, \textit{Bribe}^k_{Pj}, (), S^k, R_{Pj}(\mathcal{G}^{k+1})) \\ =& R^k_{Pj}(\textit{PubR}^k_{Pj}, \textit{Bribe}^k_{Pj}, (), S^k) + R_{Pj}(\mathcal{G}^{k+1}) \\ =& -\textit{Bribe}^k_{Pj}((), S^k) + R_{Pj}(\mathcal{G}^{k+1}) \\ \le& -r_{\min} + R_{Pj}(\mathcal{G}^{k+1}) \\ \le& 0 \end{aligned}$$

Therefore, the revenue of publisher P_j with action $(\textit{PubR}^k_{Pj}, \textit{Bribe}^k_{Pj})$ if the validator V^k does not include any report, is no higher than the revenue with action $(\textit{PubR}^k_{Pj}, \mathbf{0})$ if the validator V^k follows the Prrr protocol (Equation 23):

$$R^{k\sim m}_{Pj}(\textit{PubR}^k_{Pj}, \textit{Bribe}^k_{Pj}, (), S^k, R_{Pj}(\mathcal{G}^{k+1})) \le$$

$$R_{Pj}^{k\sim m}(\mathit{PubR}_{Pj}^k, \mathbf{0}, \boldsymbol{\mathit{PrInc}}, S^k, R_{Pj}(\mathcal{G}^{k+1}))\ .$$

(ii) When the best response of validator V^k to action $(\mathit{PubR}_{Pj}^k, \mathit{Bribe}_{Pj}^k)$ with random string S^k is not an empty vector, the revenue for publisher P_j is her revenue in step k only (Equation 16):

$$R_{Pj}^{k\sim m}(\mathit{PubR}_{Pj}^k, \mathit{Bribe}_{Pj}^k, \mathit{BR}_V(\cdot) \neq (), S^k, R_{Pj}(\mathcal{G}^{k+1})) = R_{Pj}^k(\mathit{PubR}_{Pj}^k, \mathit{Bribe}_{Pj}^k, \mathit{BR}_V(\cdot), S^k)\ .$$

Now we consider the revenue of publisher P_j with action $(\mathit{PubR}_{Pj}^k, \mathbf{0})$ if the validator V^k follows the Prrr protocol, i.e., $R_{Pj}^{k\sim m}(\mathit{PubR}_{Pj}^k, \mathbf{0}, \boldsymbol{\mathit{PrInc}}, S^k, R_{Pj}(\mathcal{G}^{k+1}))$. It is no less than her revenue in step k, $R_{Pj}^k(\mathit{PubR}_{Pj}^k, \mathbf{0}, \boldsymbol{\mathit{PrInc}}, S^k)$ (Equation 16):

$$R_{Pj}^{k\sim m}(\mathit{PubR}_{Pj}^k, \mathbf{0}, \boldsymbol{\mathit{PrInc}}, S^k, R_{Pj}(\mathcal{G}^{k+1})) \geq R_{Pj}^k(\mathit{PubR}_{Pj}^k, \mathbf{0}, \boldsymbol{\mathit{PrInc}}, S^k)\ .$$

Combining with Equation 19, we have that the revenue of publisher P_j with action $(\mathit{PubR}_{Pj}^k, \mathit{Bribe}_{Pj}^k)$ if the validator V^k follows any best response that is not an empty vector, is no higher than the revenue with action $(\mathit{PubR}_{Pj}^k, \mathbf{0})$ if the validator V^k follows the Prrr protocol:

$$R_{Pj}^{k\sim m}(\mathit{PubR}_{Pj}^k, \mathit{Bribe}_{Pj}^k, \mathit{BR}_V(\cdot) \neq (), S^k, R_{Pj}(\mathcal{G}^{k+1})) \leq R_{Pj}^{k\sim m}(\mathit{PubR}_{Pj}^k, \mathbf{0}, \boldsymbol{\mathit{PrInc}}, S^k, R_{Pj}(\mathcal{G}^{k+1}))\ .$$

Combining the cases (i) and (ii), we conclude that for any action $(\mathit{PubR}_{Pj}^k, \mathit{Bribe}_{Pj}^k)$ deviating from the Prrr protocol, the revenue of publisher P_j if the validator V^k follows any best response, is not higher than the revenue with action $(\mathit{PubR}_{Pj}^k, \mathbf{0})$ if the validator V^k follows the Prrr protocol, for any random string S^k:

$$E_{S^k}[R_{Pj}^{k\sim m}(\mathit{PubR}_{Pj}^k, \mathit{Bribe}_{Pj}^k, \mathit{BR}_V(\cdot), S^k, R_{Pj}(\mathcal{G}^{k+1}))] \leq E_{S^k}[R_{Pj}^{k\sim m}(\mathit{PubR}_{Pj}^k, \mathbf{0}, \boldsymbol{\mathit{PrInc}}, S^k, R_{Pj}(\mathcal{G}^{k+1}))]\ . \quad (25)$$

This is the first inequality in Figure 2.

Now we turn to compare the expected revenue of publisher P_j taking any action without bribery $(\mathit{PubR}_{Pj}^k, \mathbf{0})$ and following the Prrr protocol, i.e., $(\boldsymbol{\mathit{Rpt}}_{Pj}, \mathbf{0})$, when validator V^k follows the Prrr protocol (Second inequality in Figure 2).

Following the Prrr protocol, the validator action $\boldsymbol{\mathit{PrInc}}^k$ is not an empty vector. Hence, the expected revenue of publisher P_j with action $(\mathit{PubR}_{Pj}^k, \mathbf{0})$ in the subgame $\mathcal{G}^k$ only includes the expected revenue in step k:

$$R_{Pj}^{k\sim m}(\mathit{PubR}_{Pj}^k, \mathbf{0}, \boldsymbol{\mathit{PrInc}}, S^k, R_{Pj}(\mathcal{G}^{k+1})) = R_{Pj}^k(\mathit{PubR}_{Pj}^k, \mathbf{0}, \boldsymbol{\mathit{PrInc}}, S^k)\ .$$

Note that following the Prrr protocol is a special case of action $(\mathit{PubR}_{Pj}^k, \mathbf{0})$ by setting $\mathit{PubR}_{Pj}^k = \boldsymbol{\mathit{Rpt}}_{Pj}$. Therefore, the expected revenue of publisher P_j in subgame $\mathcal{G}^k$ with action $(\mathit{PubR}_{Pj}^k, \mathbf{0})$ is also her expected revenue in step k:

$$R_{Pj}^{k\sim m}(\boldsymbol{\mathit{Rpt}}_{Pj}, \mathbf{0}, \boldsymbol{\mathit{PrInc}}, S^k, R_{Pj}(\mathcal{G}^{k+1})) = R_{Pj}^k(\boldsymbol{\mathit{Rpt}}_{Pj}, \mathbf{0}, \boldsymbol{\mathit{PrInc}}, S^k)\ .$$

Due to Observation 4, the expected revenue of publisher P_j in this step with action $(\mathit{PubR}_{Pj}^k, \mathbf{0})$ is no higher than the expected revenue by following the Prrr protocol. This indicates that the expected revenue of publisher P_j in subgame $\mathcal{G}^k$ with action $(\mathit{PubR}_{Pj}^k, \mathbf{0})$ is not higher than the expected utility due to following the Prrr protocol since they are exactly the revenue in step k:

$$E_{S^k}[R_{Pj}^{k\sim m}(\mathit{PubR}_{Pj}^k, \mathbf{0}, \boldsymbol{\mathit{PrInc}}, S^k, R_{Pj}(\mathcal{G}^{k+1}))] \leq E_{S^k}[R_{Pj}^{k\sim m}(\boldsymbol{\mathit{Rpt}}_{Pj}, \mathbf{0}, \boldsymbol{\mathit{PrInc}}, S^k, R_{Pj}(\mathcal{G}^{k+1}))]\ .$$

Combined with Equation 25, we have that the expected revenue of publisher P_j in subgame $\mathcal{G}^k$ with action $(\mathit{PubR}_{Pj}^k, \mathit{Bribe}_{Pj}^k)$ and any best response of validator V^k is no higher than the expected revenue if she adopts action $(\mathit{PubR}_{Pj}^k, \mathbf{0})$ and the validator V^k follows the Prrr protocol:

$$E_{S^k}[R_{Pj}^{k\sim m}(\mathit{PubR}_{Pj}^k, \mathit{Bribe}_{Pj}^k, \mathit{BR}_V(\cdot), S^k, R_{Pj}(\mathcal{G}^{k+1}))] \leq E_{S^k}[R_{Pj}^{k\sim m}(\mathit{PubR}_{Pj}^k, \mathbf{0}, \boldsymbol{\mathit{PrInc}}, S^k, R_{Pj}(\mathcal{G}^{k+1}))]\ .$$

The utility of publisher P_j is exactly her expected revenue in subgame $\mathcal{G}^k$. Thus, the above equation shows that the utility of publisher P_j in subgame $\mathcal{G}^k$, for any deviation $(\mathit{PubR}_{Pj}^k, \mathit{Bribe}_{Pj}^k)$ and any best response of validator V^k, is no higher than her utility from following the Prrr protocol (i.e., $(\mathit{PubR}_{Pj}^k, \mathbf{0})$ with the validator following Prrr). Therefore, deviating from the Prrr protocol does not increase the utility of publisher P_j in subgame $\mathcal{G}^k$ when all other publishers follow Prrr and the validator adopts any best response. This indicates that all participants following the Prrr protocol constitutes a SPNE in subgame $\mathcal{G}^k$.

By induction, all participants following the Prrr protocol constitutes a SPNE in the entire game. □

7.3. Equilibrium Robustness

In Section 7.2, we proved that all participants following the Prrr protocol constitutes an SPNE. Now, we analyze the robustness of this equilibrium from two perspectives.

First, while the SPNE ensures that no single participant can increase their revenue by deviating, we also consider the possibility of cooperative deviations involving multiple participants. This includes two classic threats to mechanism design [58], [59] and blockchain systems [53], [60]: collusion among multiple existing participants; and Sybil attacks, where a single participant creates multiple identities and coordinates their actions to behave cooperatively. We prove that collusion or Sybil attacks of publishers (§ 7.3.1) and the collusion between publishers and a validator (§ 7.3.2) cannot increase their total expected revenue by deviating from the Prrr protocol.

Second, we analyze the stability of our SPNE (§ 7.3.3), focusing on the effects of deviations. We prove that validator deviations strictly reduce their revenue if Prrr employs the logarithmic random-value function (§ 5.5). For publishers, we demonstrate that no publisher can deviate in a way that

influences other participants' revenue without also reducing her own revenue.

7.3.1. Publisher Collusion and Sybil Attacks. We first demonstrate that Sybil-proofness can be derived from collusion-proofness among publishers. In a Sybil attack, a publisher creates multiple fake identities and distributes her reports among these identities. Then these fake identities coordinate their actions to maximize their total expected revenue. Recall that, following the Prrr protocol, a publisher's expected revenue is proportional to her share of the total reports. If a Sybil attack could increase the attacker's total expected revenue, it would imply that, if these fake identities were independent publishers, they could collude to achieve a combined revenue exceeding the proportional share of their total reports. Thus, if collusion among multiple publishers cannot increase their total expected revenue, it also ensures that Sybil attacks are not profitable.

Therefore, we focus on analyzing collusion among multiple publishers. We found that if a subset of publishers collude in the game $\mathcal{G}$, we can construct a transformed game $\mathcal{G}'$. In this transformed game, the colluding publishers are represented as a single publisher. We map the actions of the colluding publishers in $\mathcal{G}$ to the actions of this single publisher in $\mathcal{G}'$. Then the total revenue of colluding publishers in $\mathcal{G}$ equals the revenue of this single publisher in $\mathcal{G}'$, while revenues of all other participants remain unchanged.

Lemma 2. *In game $\mathcal{G}$, assuming the set of publishers is $\boldsymbol{P} = \{P_1, P_2, \ldots, P_n\}$ and their generated reports in step k are $\boldsymbol{Rpt}_1^k, \boldsymbol{Rpt}_2^k, \ldots, \boldsymbol{Rpt}_n^k$, respectively. Denote by $C \subseteq \{P_1, P_2, \ldots, P_n\}$ the colluding publishers. We now construct a new game $\mathcal{G}'$ with a single publisher P_C representing all colluding publishers:*

- *The publisher set is $\boldsymbol{P'} = \{P_C\} \cup \boldsymbol{P} \setminus C$.*
- *Publisher P_C's reports in step k are the union of all reports from colluding publishers in step k: $\boldsymbol{Rpt}_{PC}^k = \bigcup_{P_j \in C} \boldsymbol{Rpt}_{Pj}^k$.*

Considering any execution of game $\mathcal{G}$, we map the actions of participants to the new game $\mathcal{G}'$ as follows:

- *Given actions of colluding publishers in the original game $\{(\mathit{PubR}_{Pj}^k, \mathit{Bribe}_{Pj}^k)\}_{P_j \in C}$ in any step k, publisher P_C's action in step k in the new game $\mathcal{G}'$ is defined as: (1) The published report is the union set of colluders $\mathit{PubR}_{PC}^k = \bigcup_{P_j \in C} \mathit{PubR}_{Pj}^k$. (2) The output of the bribe function is the sum of all colluders' bribe functions: $\mathit{Bribe}_{PC}^k(\cdot,\cdot) = \sum_{P_j \in C} \mathit{Bribe}_{Pj}^k(\cdot,\cdot)$.*
- *Publishers other than P_C keep their reports and actions the same as in the original game $\mathcal{G}$.*

Then, for any step and random string, the revenue of the validator and all non-colluding publishers is identical in both executions, and the total revenue of the colluding publishers in $\mathcal{G}$ equals the revenue of P_C in $\mathcal{G}'$.

This lemma can be directly derived from the definition of publisher and validator revenues (Equations 14 and 15).

Proof. With the given transformation, the revenue of the validator (Equation 14) remains unchanged since the total bribe and the set of published reports from colluding publishers is preserved. The revenue of non-colluding publishers (Equation 15) also remains unchanged since it only depends on their own actions and the validator's action. We denote by $\boldsymbol{\mathit{IncR}}^k = (\mathit{IncR}_1^k, \mathit{IncR}_2^k, \ldots)$ the inclusion vector chosen by the validator in step k. The sum of the revenue of colluding publishers in the original game $\mathcal{G}$ in step k with random string S^k is

$$
\begin{aligned}
&\sum_{P_j \in C} R_{Pj}^k(\mathit{PubR}_{Pj}^k, \mathit{Bribe}_{Pj}^k, \boldsymbol{\mathit{IncR}}^k(\cdot), S^k) \\
&= \sum_{P_j \in C} \left(\begin{smallmatrix} \mathcal{I}(\mathit{IncR}_1^k \in \mathit{PubR}_{Pj}^k) \cdot \\ (\mathit{RV}(\mathit{IncR}_1^k, S^k) - \mathit{RV}(\mathit{IncR}_2^k, S^k)) \end{smallmatrix} \right) - \\
&\quad \sum_{P_j \in C} \mathit{Bribe}_{Pj}^k(\boldsymbol{\mathit{IncR}}^k, S^k) \\
&= \left(\begin{smallmatrix} \mathcal{I}(\mathit{IncR}_1^k \in \bigcup_{P_j \in C} \mathit{PubR}_{Pj}^k) \cdot \\ (\mathit{RV}(\mathit{IncR}_1^k, S^k) - \mathit{RV}(\mathit{IncR}_2^k, S^k)) \end{smallmatrix} \right) - \\
&\quad \sum_{P_j \in C} \mathit{Bribe}_{Pj}^k(\boldsymbol{\mathit{IncR}}^k, S^k) \ .
\end{aligned}
$$

This sum is exactly the revenue of publisher P_C in the new game $\mathcal{G}'$ in step k with random string S^k. □

Since in the new game $\mathcal{G}'$, all participants following the Prrr protocol constitutes a SPNE (Theorem 2), colluding publishers cannot increase their total expected revenue by deviating from the Prrr protocol in the original game $\mathcal{G}$. Therefore, Prrr is collusion-resistant among publishers. Since Sybil attacks can be reduced to collusion among multiple publishers, Prrr is also Sybil-proof.

Corollary 1. *In the SPNE of game $\mathcal{G}$ where all participants follow the Prrr protocol, colluding publishers cannot increase their total expected revenue by deviating from the Prrr protocol and a publisher cannot increase her expected revenue by launching a Sybil attack.*

7.3.2. Publisher and Validator Collusion. Now we consider collusion between a subset of publishers and the validator in a single step.

We only focus on the collusion between publishers and the validator in the first step, denoted by V^1. We do not consider the collusion with a future validator since we assume a large validator set (Section 3.4) so publishers do not know the identity of future validators [61], [62].

Since colluding publishers can be represented as a single publisher (Lemma 2), it suffices to analyze collusion between a single publisher P_j and validator V^1 in the first step. Then, we have:

Lemma 3. *In game $\mathcal{G}(RV)$ where the random-value function RV satisfies Reward Monotonicity (Property 1) and Skipping Resistance (Property 2), in the SPNE where all participants follow the Prrr protocol, collusion between any publishers and validator V^1 in the first step does not increase their total revenue.*

Suppose P_j learns the private random string S^1 before acting. She can then publish only her report *Rpt* that maximizes $RV(Rpt, S^1)$, rather than publishing all her reports. If *Rpt* is the highest among all publishers' reports, this strategy avoids the case where another of her reports is the second-highest, which would reduce her reward. However, this action decreases the validator's revenue, since she can at most get the random value of the third-highest report instead of the second-highest. Thus, while the publisher may increase her own reward, the total revenue for the colluding pair (publisher and validator) does not increase compared to following the protocol. In other words, collusion between a publisher and the validator in the first step cannot increase their combined revenue.

Proof. First, we present the revenue structure of the colluding parties. Next, we analyze their revenue when they adhere to the Prrr protocol. Finally, we calculate their revenue with deviation and compare it to the revenue obtained by following the Prrr protocol.

(1) Revenue Structure of Colluding Parties:

Since colluding publishers can be represented as a single publisher (Lemma 2), it suffices to analyze collusion between a single publisher P_j and validator V^1 in the first step. If P_j learns the private random string S^1 before acting, her revenue in the game (Equation 16) consists of her revenue in step 1 (given S^1) plus her expected revenue in the subgame $\mathcal{G}^2$ if validator V^1 does not include any report. Recall that $R_{Pj}(\mathcal{G}^2)$ is the expected revenue for publisher P_j in the subgame starting from step 2. We denote publisher P_j's action by $(PubR^1_{Pj}, Bribe^1_{Pj})$ and validator V^1's action by $\boldsymbol{IncR}^1 = (IncR^1_1, IncR^1_2, \ldots)$. The revenue of publisher P_j (Equation 16) in the game is:

$$\begin{aligned}
&R^{1\sim m}_{Pj}(PubR^1_{Pj}, Bribe^1_{Pj}, \boldsymbol{IncR}^1, S^k, R_{Pj}(\mathcal{G}^2))\\
=&R^1_{Pj}(PubR^1_{Pj}, Bribe^1_{Pj}, \boldsymbol{IncR}^1, S^1)+\\
&\mathcal{I}(\boldsymbol{IncR}^1 = ()) \cdot R_{Pj}(\mathcal{G}^2)\\
\overset{Eq.15}{=}&\begin{pmatrix}\mathcal{I}(IncR^1_1 \in PubR^1_{Pj})\cdot\\ (RV(IncR^1_1, S^1) - RV(IncR^1_2, S^1))\end{pmatrix} - Bribe^1_{Pj}(\boldsymbol{IncR}^1, S^1)+\\
&\mathcal{I}(\boldsymbol{IncR}^1 = ()) \cdot R_{Pj}(\mathcal{G}^2)\ . \qquad (26)
\end{aligned}$$

Recall that $\boldsymbol{PubR}^1 = (PubR^1_{P1}, \ldots, PubR^1_{Pn})$ and that $\boldsymbol{Bribe}^1 = (Bribe^1_{P1}, \ldots, Bribe^1_{Pn})$ are the actions of all publishers in step 1. Note that all publishers other than P_j follow the Prrr protocol, i.e., $PubR^1_{Pi} = \boldsymbol{Rpt}_{Pi}$ and $Bribe^1_{Pi} = \mathbf{0}$ for all $P_i \neq P_j$. The revenue of validator V^1 (Equation 14) with actions $\boldsymbol{PubR}^1, \boldsymbol{Bribe}^1$ and $\boldsymbol{IncR}^1$ is:

$$\begin{aligned}
&R^1_V(\boldsymbol{IncR}^1, \boldsymbol{Bribe}^1, S^1)\\
=&\sum_i Bribe^1_{Pi}(\boldsymbol{IncR}^1, S^1)+\\
&\mathcal{I}(|\boldsymbol{IncR}^1| = 2) \cdot RV(IncR^1_2, S^1)\\
=&Bribe^1_{Pj}(\boldsymbol{IncR}^1, S^1) + \mathcal{I}(|\boldsymbol{IncR}^1| = 2) \cdot RV(IncR^1_2, S^1)\ . \qquad (27)
\end{aligned}$$

Denote by $R_{collude}$ the total revenue of publisher P_j and validator V^1 (Equation 14) due to the above actions:

$$\begin{aligned}
&R_{collude}(\boldsymbol{PubR}^1, \boldsymbol{Bribe}^1, \boldsymbol{IncR}^1, S^k, R_{Pj}(\mathcal{G}^2))\\
=&R^{1\sim m}_{Pj}(PubR^1_{Pj}, Bribe^1_{Pj}, \boldsymbol{IncR}^1, S^k, R_{Pj}(\mathcal{G}^2))+\\
&R^1_V(\boldsymbol{IncR}^1, \boldsymbol{Bribe}^1, S^1)\\
\overset{Eq.26}{=}&\begin{pmatrix}\mathcal{I}(IncR^1_1 \in PubR^1_{Pj})\cdot\\ (RV(IncR^1_1, S^1) - RV(IncR^1_2, S^1))\end{pmatrix} - Bribe^1_{Pj}(\boldsymbol{IncR}^1, S^1)+\\
&\mathcal{I}(\boldsymbol{IncR}^1 = ()) \cdot R_{Pj}(\mathcal{G}^2) + R^1_V(\boldsymbol{IncR}^1, \boldsymbol{Bribe}^1, S^1)\\
\overset{Eq.27}{=}&\begin{pmatrix}\mathcal{I}(IncR^1_1 \in PubR^1_{Pj})\cdot\\ (RV(IncR^1_1, S^1) - RV(IncR^1_2, S^1))\end{pmatrix} - Bribe^1_{Pj}(\boldsymbol{IncR}^1, S^1)+\\
&\mathcal{I}(\boldsymbol{IncR}^1 = ()) \cdot R_{Pj}(\mathcal{G}^2) + Bribe^1_{Pj}(\boldsymbol{IncR}^1, S^1)+\\
&\mathcal{I}(|\boldsymbol{IncR}^1| = 2) \cdot RV(IncR^1_2, S^1)\\
=&\begin{pmatrix}\mathcal{I}(IncR^1_1 \in PubR^1_{Pj})\cdot\\ (RV(IncR^1_1, S^1) - RV(IncR^1_2, S^1))\end{pmatrix} +\\
&\mathcal{I}(\boldsymbol{IncR}^1 = ()) \cdot R_{Pj}(\mathcal{G}^2)+\\
&\mathcal{I}(|\boldsymbol{IncR}^1| = 2) \cdot RV(IncR^1_2, S^1)\ . \qquad (28)
\end{aligned}$$

For notation simplicity, we use $R_{collude}(PubR^1_{Pj}, Bribe^1_{Pj}, \boldsymbol{IncR}^1, S^k, R_{Pj}(\mathcal{G}^2))$ instead of $R_{collude}(\boldsymbol{PubR}^1, \boldsymbol{Bribe}^1, \boldsymbol{IncR}^1, S^k, R_{Pj}(\mathcal{G}^2))$, since all other publishers follow the Prrr protocol.

(2) Revenue When Following the Prrr Protocol:

When both publisher P_j and validator V^1 follow the Prrr protocol, the action of validator V^1 is $\boldsymbol{PrInc} = (IncR^1_1, IncR^1_2)$ and the action of publisher P_j is $(\boldsymbol{Rpt}_{Pj}, \mathbf{0})$. The total revenue of publisher P_j and validator V^1 by following the Prrr protocol is:

$$\begin{aligned}
&R_{collude}(\boldsymbol{Rpt}_{Pj}, \mathbf{0}, \boldsymbol{PrInc}, S^k, R_{Pj}(\mathcal{G}^2)) =\\
&\mathcal{I}(IncR^1_1 \in PubR^1_{Pj}) \cdot (RV(IncR^1_1, S^1) - RV(IncR^1_2, S^1))+\\
&\qquad\qquad RV(IncR^1_2, S^1)
\end{aligned}$$

When publisher P_j holds the report with the highest value, then $IncR^1_1 \in \boldsymbol{Rpt}_{Pj}$, so

$$\begin{aligned}
&\mathcal{I}(IncR^1_1 \in PubR^1_{Pj}) \cdot (RV(IncR^1_1, S^1) - RV(IncR^1_2, S^1)) =\\
&\qquad (RV(IncR^1_1, S^1) - RV(IncR^1_2, S^1)),
\end{aligned}$$

otherwise, $\mathcal{I}(IncR^1_1 \in PubR^1_{Pj}) = 0$. Therefore, the total revenue of publisher P_j and validator V^1 when following the Prrr protocol is:

$$\begin{aligned}
&R_{collude}(\boldsymbol{Rpt}_{Pj}, \mathbf{0}, \boldsymbol{PrInc}, S^k, R_{Pj}(\mathcal{G}^2)) =\\
&\begin{cases}
\max\limits_{Rpt \in \bigcup_i \boldsymbol{Rpt}_{Pi}} RV(Rpt, S^1),\\
\quad \text{if} \quad \underset{Rpt \in \bigcup_i \boldsymbol{Rpt}_{Pi}}{argmax}\ RV(Rpt, S^1) \in \boldsymbol{Rpt}_{Pj};\\
\\
\max\limits^{(2)}_{Rpt \in \bigcup_i \boldsymbol{Rpt}_{Pi}} RV(Rpt, S^1), \qquad \text{otherwise.}
\end{cases} \qquad (29)
\end{aligned}$$

(3) Revenue with Deviation:

We now analyze the revenue of the colluding parties when they deviate from the Prrr protocol. We separate into two cases based on whether validator V^1 includes any report.

Case (i): Validator V^1 includes no report, i.e., $\boldsymbol{IncR}^1 = ()$.

In this case, the total revenue (Equation 28) of publisher P_j and validator V^1 is $R_{Pj}(\mathcal{G}^2)$. Since there is no colluding validator in subgame $\mathcal{G}^2$, all participants following the Prrr protocol in subgame $\mathcal{G}^2$ constitute an SPNE(Theorem 2). The deviation of publisher P_j does not increase her expected revenue in subgame $\mathcal{G}^2$. Therefore, the revenue of publisher P_j in subgame $\mathcal{G}^2$ is no higher than her revenue by following the Prrr protocol. Recall that by following the Prrr protocol in subgame $\mathcal{G}^2$, the revenue of publisher is proportional to the ratio of her report count to the total report count, multiplied by the total expected revenue (Observation 4). Thus, we have:

$$R_{Pj}(\mathcal{G}^2) \leq \frac{|\boldsymbol{Rpt}_{Pj}|}{\sum_{1\leq i\leq n}|\boldsymbol{Rpt}_{Pi}|} \cdot \mathit{RAllPub}\left(\sum_{1\leq i\leq n}|\boldsymbol{Rpt}_{Pi}|\right) .$$

Due to Skipping Resistance (Property 2), the total reward for all publishers $\mathit{RAllPub}\left(\sum_{1\leq i\leq n}|\boldsymbol{Rpt}_{Pi}|\right)$ is upper bounded by the minimum value of the reports. Consequently, the revenue of publisher P_j in subgame $\mathcal{G}^2$ cannot exceed the total revenue of the colluding parties when they follow the Prrr protocol (Equation 29). Therefore, in this case, the combined revenue of publisher P_j and validator V^1 with deviation is no higher than their revenue when adhering to the Prrr protocol.

Case (ii): Validator V^1 includes at least one report.

We analyze two sub-cases based on whether validator V^1 includes a report from publisher P_j as the first report.

Sub-case (ii.a): Validator V^1 includes a report from publisher P_j as the first report, i.e., $\mathit{IncR}_1^1 \in \mathit{PubR}_{Pj}^1$.

In this sub-case, the total revenue (Equation 28) of publisher P_j and validator V^1 is no higher than the private value of P_j's report included as the first report:

$$\begin{aligned}
&R_{\mathit{collude}}(\mathit{PubR}_{Pj}^1, \mathit{Bribe}_{Pj}^1, \boldsymbol{IncR}^1, S^1, R_{Pj}(\mathcal{G}^2)) \\
=&(\mathit{RV}(\mathit{IncR}_1^1, S^1) - \mathit{RV}(\mathit{IncR}_2^1, S^1)) + \\
&\mathcal{I}(|\boldsymbol{IncR}^1| = 2) \cdot \mathit{RV}(\mathit{IncR}_2^1, S^1) \\
\leq&\mathit{RV}(\mathit{IncR}_1^1, S^1) .
\end{aligned}$$

Hence, the total revenue of the colluders is no higher than the maximum private value of P_j's report with random string S^1:

$$R_{\mathit{collude}}(\mathit{PubR}_{Pj}^1, \mathit{Bribe}_{Pj}^1, \boldsymbol{IncR}^1, S^1, R_{Pj}(\mathcal{G}^2)) \leq \max_{\mathit{Rpt}\in\boldsymbol{Rpt}_{Pj}} \mathit{RV}(\mathit{Rpt}, S^1) .$$

We now turn back to the revenue of colluders when following the Prrr protocol (Equation 29). If publisher P_j holds the report with the highest value among all publishers, then the total revenue of colluders by following the Prrr protocol is the maximum private value of P_j's report with random string S^1. Otherwise, the total revenue of colluders by following the Prrr protocol is the second-highest private value among all publishers, which is no less than the maximum private value of P_j's report. Therefore, in this sub-case, the combined revenue of publisher P_j and validator V^1 with deviation is no higher than their revenue when adhering to the Prrr protocol.

Sub-case (ii.b): Validator V^1 does not include any report from publisher P_j as the first report, i.e., $\mathit{IncR}_1^1 \notin \mathit{PubR}_{Pj}^1$.

In this sub-case, the total revenue (Equation 28) of publisher P_j and validator V^1 is no higher than the private value of the second report included by validator V^1:

$$\begin{aligned}
&R_{\mathit{collude}}(\mathit{PubR}_{Pj}^1, \mathit{Bribe}_{Pj}^1, \boldsymbol{IncR}^1, S^1, R_{Pj}(\mathcal{G}^2)) \\
=&\mathcal{I}(|\boldsymbol{IncR}^1| = 2) \cdot \mathit{RV}(\mathit{IncR}_2^1, S^1) \\
\leq&\mathit{RV}(\mathit{IncR}_2^1, S^1) .
\end{aligned}$$

When there are multiple reports with the same highest value, the revenue of colluders with deviation is no higher than that highest value. At the same time, the revenue of colluders by following the Prrr protocol (Equation 29) is exactly that highest value since the highest and second-highest values are equal.

When there is only one report with the highest value, if the validator V^1 includes that report as the second report, then the random value of the second report is higher than the first. Recall that in Section 6.1, we introduce a dummy report with the minimum random value as the second report in the above case. Consequently, the revenue of the colluding parties with deviation is at most the second-highest private value among all publishers. However, the revenue of the colluding parties when following the Prrr protocol (Equation 29) is at least this second-highest value. Thus, in this sub-case, the combined revenue of publisher P_j and validator V^1 with deviation is no higher than their revenue when adhering to the Prrr protocol.

(4) Conclusion: In all, we have shown that in all possible deviation cases, the combined revenue of publisher P_j and validator V^1 with deviation is no higher than their revenue when adhering to the Prrr protocol. Thus, collusion between any publishers and validator V^1 in the first step does not increase their total revenue. □

7.3.3. Equilibrium Stability. Indeed, we saw (§7.2) that Prrr is a (non-strict) SPNE. We prove that validator deviations do strictly reduce their revenue if Prrr employs the logarithmic random-value function (§ C). We now show that if a publisher changes (either increases or decreases) the revenue of the validator or of other publishers, her own revenue strictly decreases.

Publishers may also have multiple best responses when all participants follow the Prrr protocol. We prove that no publisher can deviate in a way that changes the revenue of other publishers unless the deviation also reduces their own revenue.

Lemma 4. *In the game $\mathcal{G}(RV)$, assuming the random-value function RV satisfies Reward Monotonicity (Property 1) and Skipping Resistance (Property 2). Suppose all publishers follow the Prrr protocol except for a publisher P_j who deviates by taking action $(\mathit{PubR}_{Pj}^k, \mathit{Bribe}_{Pj}^k)$ in step k. Assume that if following the Prrr protocol is a best response*

for a validator V^k then she follows the Prrr protocol. If the revenue of any validator or any other publisher changes because publisher P_j deviates, then the expected revenue of publisher P_j strictly decreases.

We first show that if the best response of the validator to the deviation is to include no report, then the expected revenue of the deviating publisher strictly decreases. Next, we assume that the best response of the validator is to include some report. We prove that if the publisher P_j does not publish all her reports, then her expected revenue strictly decreases (Lemma 1). Finally, in the case where publisher P_j publishes all her reports but the revenue of any other participants changes, we prove that the expected revenue of publisher P_j strictly decreases.

Proof. Since if all participants follow the Prrr protocol the game ends at the first step, we primarily focus on the first step. We denote by $(\mathit{PubR}^1_{Pj}, \mathit{Bribe}^1_{Pj})$ the deviating action of publisher P_j in step 1. We analyze two cases based on whether validator V^1's best response is to include any report.

Before diving into the cases, we first build up some common notation and results.

Recall that $R_{Pj}(\mathcal{G}^2)$ is the expected revenue for publisher P_j in the subgame starting from step 2. Since following Prrr constitutes an SPNE in all subgames (Theorem 2), the deviation of publisher P_j does not increase her expected revenue in subgame $\mathcal{G}^2$. Therefore, the revenue of publisher P_j in subgame $\mathcal{G}^2$ is no higher than the total expected revenue of publishers if they follow the Prrr protocol, that is:

$$R_{Pj}(\mathcal{G}^2) \leq \mathit{RAllPub}\left(\sum_{1\leq i\leq n} |\boldsymbol{\mathit{Rpt}}_{Pi}|\right) \tag{30}$$

Now, we analyze the revenue of publisher P_j with deviation if the best response of validator V^1 is to include some reports, that is, $\mathit{BR}^1_V(\cdot) \neq ()$. Under this condition, given the random string S^1, the revenue of publisher P_j in the game with deviation (Equation 16) is just her revenue in step 1:

$$R^{1\sim m}_{Pj}(\mathit{PubR}^1_{Pj}, \mathit{Bribe}^1_{Pj}, \boldsymbol{\mathit{IncR}}^1 \neq (), S^1, R_{Pj}(\mathcal{G}^2)) = R^1_{Pj}(\mathit{PubR}^1_{Pj}, \mathit{Bribe}^1_{Pj}, \boldsymbol{\mathit{IncR}}^1, S^1)\ .$$

By Lemma 1, for any random string S^1, the revenue of publisher P_j in step 1 with deviation is at most her revenue when she takes action $(\boldsymbol{\mathit{Rpt}}_{Pj}, \mathbf{0})$ and validator V^1 follows the Prrr protocol. Since her total revenue in the game is just her revenue in step 1, it follows that with any deviating action $(\mathit{PubR}^1_{Pj}, \mathit{Bribe}^1_{Pj})$, her revenue is no higher than if she takes action $(\mathit{PubR}^1_{Pj}, \mathbf{0})$ and the validator follows the Prrr protocol:

$$R^{1\sim m}_{Pj}(\mathit{PubR}^1_{Pj}, \mathit{Bribe}^1_{Pj}, \boldsymbol{\mathit{IncR}}^1 \neq (), S^1, R_{Pj}(\mathcal{G}^2)) \leq R^{1\sim m}_{Pj}(\mathit{PubR}^1_{Pj}, \mathbf{0}, \boldsymbol{\mathit{PrInc}}, S^1, R_{Pj}(\mathcal{G}^2))\ . \tag{31}$$

Now we consider the two cases separately.

Case (1): There is a random string S^1 where validator V^1's best response to the deviation is to include no report.

In this case, we prove that the expected revenue of publisher P_j with deviation is strictly less than her expected revenue by following the Prrr protocol.

We compare the revenue of publisher P_j with deviation to the revenue she would obtain by the same set of reports without bribes, i.e., $(\mathit{PubR}^1_{Pj}, \mathbf{0})$. It suffices to show that her revenue with deviation if the validator follows any best response is strictly less than her revenue with action $(\mathit{PubR}^1_{Pj}, \mathbf{0})$ if the validator follows the Prrr protocol; in this case, the deviation cannot be a best response. Note that with P_j's action $(\mathit{PubR}^1_{Pj}, \mathbf{0})$, following the Prrr protocol is a best response for validator V^1.

If the best response of the validator is to include no report, then the revenue of the validator only comes from bribes (Equation 14). Since following the Prrr protocol has at least $r_{\min}$ revenue for the validator, the total bribe offered by all publishers must be at least $r_{\min}$. Thus, the publisher P_j must pay at least $r_{\min}$ bribe. Therefore, her revenue in the game $\mathcal{G}$ is most $R_{Pj}(\mathcal{G}^2) - r_{\min}$.

Combining Skipping Resistance (Property 2) and Equation 30, we have $R_{Pj}(\mathcal{G}^2) - r_{\min} < 0$. Therefore, P_j's revenue in this case is strictly less than 0, which is also strictly less than her revenue when she takes action $(\mathit{PubR}^1_{Pj}, \mathbf{0})$ and the validator follows the Prrr protocol:

$$R^{1\sim m}_{Pj}(\mathit{PubR}^1_{Pj}, \mathit{Bribe}^1_{Pj}, \boldsymbol{\mathit{IncR}}^1 = (), S^1, R_{Pj}(\mathcal{G}^2)) < R^{1\sim m}_{Pj}(\mathit{PubR}^1_{Pj}, \mathbf{0}, \boldsymbol{\mathit{PrInc}}, S^1, R_{Pj}(\mathcal{G}^2))$$

Recall that if the best response of the validator is to include some reports, the revenue of publisher P_j with deviation is at most her revenue when she takes action $(\mathit{PubR}^1_{Pj}, \mathbf{0})$ and the validator follows the Prrr protocol (Equation 31). Therefore, if there exists any random string S^1 for which the validator's best response is to include no report, the expected revenue of publisher P_j with deviation is strictly less than her expected revenue when taking action $(\mathit{PubR}^1_{Pj}, \mathbf{0})$ and the validator follows the Prrr protocol. Thus, such a deviation cannot be a best response for publisher P_j.

Case (2): For all random strings S^1, validator V^1's best response to the deviation is to include some reports.

We separate to two subcases, based on whether the publisher P_j publishes all of her reports.

Subcase (i), Publisher P_j does not publish all of her reports, i.e., $\mathit{PubR}^1_{Pj} \neq \boldsymbol{\mathit{Rpt}}_{Pj}$:

In this subcase, we prove that the expected revenue of publisher P_j with deviation is strictly less than her expected revenue by following the Prrr protocol.

Recall that in this case including an empty set of reports is not the best response of validator V^1. Therefore, the revenue of publisher P_j with deviation is at most her revenue when she takes action $(\mathit{PubR}^1_{Pj}, \mathbf{0})$ and the validator follows the Prrr protocol (Equation 31). Therefore the expected revenue of publisher P_j with deviation if the validator follows any best response is at most her expected revenue when

taking action $(\mathit{PubR}^1_{Pj}, \mathbf{0})$ and the validator follows the Prrr protocol. Combined with Observation 3, we have:

$$\begin{aligned}
&\mathbb{E}_{S^1}\left[R^{1\sim m}_{Pj}(\mathit{PubR}^1_{Pj}, \mathit{Bribe}^1_{Pj}, \boldsymbol{\mathit{IncR}}^1 \neq (), S^1, R_{Pj}(\mathcal{G}^2))\right] \\
\leq &\mathbb{E}_{S^1}\left[R^{1\sim m}_{Pj}(\mathit{PubR}^1_{Pj}, \mathbf{0}, \boldsymbol{\mathit{PrInc}}, S^1, R_{Pj}(\mathcal{G}^2))\right] \\
= &\frac{|\mathit{PubR}^1_{Pj}| \cdot \mathit{RAllPub}\left(|\mathit{PubR}^1_{Pj}| + \sum_{i\neq j} |\boldsymbol{\mathit{Rpt}}_{Pi}|\right)}{|\mathit{PubR}^1_{Pj}| + \sum\limits_{i\neq j} |\boldsymbol{\mathit{Rpt}}_{Pi}|} \ .
\end{aligned}$$

While by following the Prrr protocol, the expected revenue of publisher P_j is:

$$\begin{aligned}
&\mathbb{E}_{S^1}\left[R^{1\sim m}_{Pj}(\boldsymbol{\mathit{Rpt}}_{Pj}, \mathbf{0}, \boldsymbol{\mathit{PrInc}}, S^1, R_{Pj}(\mathcal{G}^2))\right] = \\
&\qquad \frac{|\boldsymbol{\mathit{Rpt}}_{Pj}|}{\sum_{1\leq i\leq n} |\boldsymbol{\mathit{Rpt}}_{Pi}|} \cdot \mathit{RAllPub}\left(\sum_{1\leq i\leq n} |\boldsymbol{\mathit{Rpt}}_{Pi}|\right) \ .
\end{aligned}$$

Since $|\mathit{PubR}^1_{Pj}| < |\boldsymbol{\mathit{Rpt}}_{Pj}|$, due to Reward Monotonicity (Property 1), we have:

$$\begin{aligned}
&\mathit{RAllPub}\left(|\mathit{PubR}^1_{Pj}| + \sum_{i\neq j} |\boldsymbol{\mathit{Rpt}}_{Pi}|\right) \leq \\
&\qquad \mathit{RAllPub}\left(\sum_{1\leq i\leq n} |\boldsymbol{\mathit{Rpt}}_{Pi}|\right) \ .
\end{aligned}$$

We also trivially have:

$$\frac{|\mathit{PubR}^1_{Pj}|}{|\mathit{PubR}^1_{Pj}| + \sum\limits_{i\neq j} |\boldsymbol{\mathit{Rpt}}_{Pi}|} < \frac{|\boldsymbol{\mathit{Rpt}}_{Pj}|}{\sum_{1\leq i\leq n} |\boldsymbol{\mathit{Rpt}}_{Pi}|} \ .$$

Therefore, the expected revenue of publisher P_j with deviation is strictly less than her expected revenue by following the Prrr protocol.

Subcase (ii), Publisher P_j publishes all of her reports, i.e., $\mathit{PubR}^1_{Pj} = \boldsymbol{\mathit{Rpt}}_{Pj}$:

In previous case and subcase, we have only shown that the expected revenue of publisher P_j with deviation is strictly less than her expected revenue by following the Prrr Protocol. In this subcase, we study whether the deviation of publisher P_j affects the revenue of other participants.

First, the revenue of validators after step 1 remains unaffected since validator V^1 includes some reports in step 1 (condition of Case (2)). The revenue of validators after step 1 consistently remains zero, regardless of whether a deviation occurs.

Second, if the deviating publisher P_j could increase the revenue of the first validator V^1 without reducing her own revenue, it would imply that their collusion increases their total revenue. This contradicts Lemma 3. On the other hand, since P_j still publishes all reports in the deviation (as per the condition of this subcase), the validator V^1, by following the Prrr protocol, would receive the bribe from P_j plus the second-highest value. This is no less than the revenue obtained by following the Prrr protocol when P_j does not deviate. Thus, P_j cannot reduce the revenue of validator V^1. In conclusion, P_j cannot alter validator V^1's revenue without reducing her own revenue.

Therefore, the remaining task is to prove that the revenue of other publishers would not be affected without reducing P_j's revenue.

To change the revenue of another publisher $P_{j\prime}$, the deviation of publisher P_j must change the revenue of $P_{j\prime}$ with at least one random string S^1. We prove that, in this case, the revenue of publisher P_j with deviation is strictly less than her revenue when following the Prrr protocol.

To make following the Prrr protocol not a best response for validator V^1 with random string S^1, publisher P_j needs to at least provide a positive bribe. Therefore, if the best response of validator V^1 to deviation does not include the report from publisher P_j as the first report, P_j's revenue with deviation is strictly less than zero. Thus, if P_j's revenue does not decrease, the best response of validator V^1 to deviation must include the report from publisher P_j as the first report.

If validator V^1 does not include $P_{j\prime}$'s report as the first report when all participants follow the Prrr protocol, then $P_{j\prime}$'s revenue is zero, which remains unchanged with any deviation. Therefore, for $P_{j\prime}$'s revenue to change due to a deviation, validator V^1 must include $P_{j\prime}$'s report as the first report when all participants follow the Prrr protocol. This implies that $P_{j\prime}$'s report has the unique highest value among all reports; otherwise, if the highest value is not unique, $P_{j\prime}$'s revenue is zero according to the Prrr protocol, contradicting the assumption that her revenue changes with deviation.

Therefore, we assume that without deviation, validator V^1 includes the report from publisher $P_{j\prime}$ as the first report.

Recall that $\max^{(2)}_{\mathit{Rpt}\in\bigcup_i \boldsymbol{\mathit{Rpt}}_{Pi}} \mathit{RV}(\mathit{Rpt}, S^1)$ is the second-highest value among all reports with random string S^1. The revenue of the validator when all participants follow the Prrr protocol is exactly the value of this second-highest report:

$$R^1_V(\boldsymbol{\mathit{PrInc}}, \mathbf{0}, S^1) = \max^{(2)}_{\mathit{Rpt}\in\bigcup_i \boldsymbol{\mathit{Rpt}}_{Pi}} \mathit{RV}(\mathit{Rpt}, S^1) \ .$$

Denote by r_1 and r_2 the private values of the reports validator V^1 includes as the first and second reports to deviation, respectively. The revenue of validator V^1 with deviation is r_2 plus the bribe from publisher P_j:

$$R^1_V(\mathit{BR}^1_V(\cdot), \mathit{Bribe}^1_{Pj}, S^1) = r_2 + \mathit{Bribe}^1_{Pj}(\mathit{BR}^1_V(\cdot), S^1) \ .$$

This revenue must be strictly higher than the revenue when all participants follow the Prrr protocol. Otherwise, following the Prrr protocol would still be a best response for validator V^1 with deviation. Therefore, we have the lower bound on the bribe offered by publisher P_j:

$$\mathit{Bribe}^1_{Pj}(\mathit{BR}^1_V(\cdot), S^1) > \max^{(2)}_{\mathit{Rpt}\in\bigcup_i \boldsymbol{\mathit{Rpt}}_{Pi}} \mathit{RV}(\mathit{Rpt}, S^1) - r_2 \ . \tag{32}$$

The revenue of publisher P_j with deviation is:

$$\begin{aligned}
&R^1_{Pj}(\boldsymbol{\mathit{Rpt}}_{Pj}, \mathit{Bribe}^1_{Pj}, \boldsymbol{\mathit{IncR}}^1, S^1) \\
=&(r_1 - r_2) - \mathit{Bribe}^1_{Pj}(\boldsymbol{\mathit{IncR}}^1)
\end{aligned}$$

$$\overset{Eq.32}{<} (r_1 - r_2) - \left(\max_{Rpt \in \bigcup_i \boldsymbol{Rpt}_{Pi}}^{(2)} RV(Rpt, S^1) - r_2 \right)$$
$$= r_1 - \max_{Rpt \in \bigcup_i \boldsymbol{Rpt}_{Pi}}^{(2)} RV(Rpt, S^1) \ .$$

Since r_1 is a report from publisher P_j but not the report with the highest value, we have

$$r_1 \leq \max_{Rpt \in \bigcup_i \boldsymbol{Rpt}_{Pi}}^{(2)} RV(Rpt, S^1).$$

Therefore, the revenue of publisher P_j with deviation with S^1 is strictly negative. This implies that any such deviation strictly reduces her revenue compared to following the Prrr protocol.

From Equation 31, with any random string S^1, publisher P_j's revenue with any deviating action $(\boldsymbol{Rpt}_{Pj}, Bribe^1_{Pj})$ if the validator follows any best response is not higher than her revenue when she follows the Prrr protocol. Therefore, if there exists any random string S^1 where the revenue of publisher P_j with deviation is strictly less than her revenue when following the Prrr protocol, then the deviation is not the best response for the publisher.

Conclusion In all, considering all cases, we have shown that if the deviation of publisher P_j changes the revenue of any other publisher, then the expected revenue of publisher P_j strictly decreases. □

8. Conclusion

We present Prrr, a novel protocol for the prevalent blockchain reporting problem that addresses the security-performance trade-off. Prrr utilizes a mechanism-design concept called EASA, which is the first game-theoretic concept to deliberately form participant asymmetry. We prove that all participants following the Prrr protocol constitutes a subgame perfect Nash equilibrium (SPNE) and this SPNE is robust against collusion and Sybil attacks. Prrr is directly applicable to the numerous smart contracts that rely on timely reports.

Acknowledgments

This work was supported in part IC3 and the Avalanche Foundation.

References

[1] O. Knight, "Defi tvl rebounds to $170b, erasing terra-era bear market losses," https://www.coindesk.com/business/2025/09/18/defi-tvl-rebounds-to-usd170b-erasing-terra-era-bear-market-losses, 2025, accessed, November 2025.

[2] M. Moosavi, M. Salehi, D. Goldman, and J. Clark, "Fast and furious withdrawals from optimistic rollups," in *5th Conference on Advances in Financial Technologies (AFT 2023)*. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2023, pp. 22–1.

[3] S. Motepalli, L. Freitas, and B. Livshits, "Sok: Decentralized sequencers for rollups," *arXiv preprint arXiv:2310.03616*, 2023.

[4] W. Wang, L. Zhou, A. Yaish, F. Zhang, B. Fisch, and B. Livshits, "Mechanism design for zk-rollup prover markets," *arXiv preprint arXiv:2404.06495*, 2024.

[5] A. Zamyatin, D. Harz, J. Lind, P. Panayiotou, A. Gervais, and W. Knottenbelt, "Xclaim: Trustless, interoperable, cryptocurrency-backed assets," in *2019 IEEE symposium on security and privacy (SP)*. IEEE, 2019, pp. 193–210.

[6] M. Herlihy, "Atomic cross-chain swaps," in *Proceedings of the 2018 ACM symposium on principles of distributed computing*, 2018, pp. 245–254.

[7] T. Xie, J. Zhang, Z. Cheng, F. Zhang, Y. Zhang, Y. Jia, D. Boneh, and D. Song, "zkbridge: Trustless cross-chain bridges made practical," in *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, 2022, pp. 3003–3017.

[8] A. Miller, I. Bentov, S. Bakshi, R. Kumaresan, and P. McCorry, "Sprites and state channels: Payment networks that go faster than lightning," in *International conference on financial cryptography and data security*. Springer, 2019, pp. 508–526.

[9] L. Gudgeon, P. Moreno-Sanchez, S. Roos, P. McCorry, and A. Gervais, "Sok: Layer-two blockchain protocols," in *International Conference on Financial Cryptography and Data Security*. Springer, 2020, pp. 201–226.

[10] M. Khabbazian, T. Nadahalli, and R. Wattenhofer, "Outpost: A responsive lightweight watchtower," in *Proceedings of the 1st ACM Conference on Advances in Financial Technologies*, 2019, pp. 31–40.

[11] B. Liu, P. Szalachowski, and S. Sun, "Fail-safe watchtowers and short-lived assertions for payment channels," in *Proceedings of the 15th ACM Asia Conference on Computer and Communications Security*, 2020, pp. 506–518.

[12] Z. Avarikioti, O. S. Thyfronitis Litos, and R. Wattenhofer, "Cerberus channels: Incentivizing watchtowers for bitcoin," in *International Conference on Financial Cryptography and Data Security*. Springer, 2020, pp. 346–366.

[13] Z. He, J. Li, and Z. Wu, "Don't trust, verify: The case of slashing from a popular ethereum explorer," in *Companion Proceedings of the ACM Web Conference 2023*, 2023, pp. 1078–1084.

[14] K. Qin, L. Zhou, P. Gamito, P. Jovanovic, and A. Gervais, "An empirical study of defi liquidations: Incentives, risks, and instabilities," in *Proceedings of the 21st ACM internet measurement conference*, 2021, pp. 336–350.

[15] M. Morini, "Managing derivatives on a blockchain. a financial market professional implementation," *A Financial Market Professional Implementation (May 5, 2017)*, 2017.

[16] M. Dai, L. Li, and C. Yang, "Arbitrage in perpetual contracts," *Available at SSRN 5262988*, 2025.

[17] E. Chen, M. Ma, and Z. Nie, "Perpetual future contracts in centralized and decentralized exchanges: Mechanism and traders' behavior," *Electronic Markets*, vol. 34, no. 1, p. 35, 2024.

[18] B. Lindrea, "Base blames faulty sequencer for 33-minute outage, fixes made," https://cointelegraph.com/news/base-blames-faulty-sequencer-33-minute-network-outage, 2025, accessed, October 2025.

[19] S. D. Young, "Arbitrum temporarily stopped processing due to software bug," https://www.coindesk.com/tech/2023/06/07/arbitrum-temporarily-stopped-processing-due-to-software-bug, 2023, accessed, October 2025.

[20] Z. Várdai, "Ethereum l2 starknet suffers 2nd mainnet outage in 2 months," https://cointelegraph.com/news/starknet-outage-ethereum-l2-reliability-concerns, 2025, accessed, October 2025.

[21] P. Daian, S. Goldfeder, T. Kell, Y. Li, X. Zhao, I. Bentov, L. Breidenbach, and A. Juels, "Flash boys 2.0: Frontrunning in decentralized exchanges, miner extractable value, and consensus instability," in *2020 IEEE symposium on security and privacy (SP)*. IEEE, 2020, pp. 910–927.

[22] Cointelegraph, "Fallout: Ethereum fees skyrocketed as traders raced to unwind leveraged positions," https://cointelegraph.com/news/fallout-ethereum-fees-skyrocketed-as-traders-raced-to-unwind-leveraged-positions, 2021, accessed, October 2025.

[23] I. Tsabary, M. Yechieli, A. Manuskin, and I. Eyal, "Mad-htlc: because htlc is crazy-cheap to attack," in *2021 IEEE symposium on security and privacy (SP)*. IEEE, 2021, pp. 1230–1248.

[24] L. S. Brugger, L. Kovács, A. Petkovic Komel, S. Rain, and M. Rawson, "Checkmate: Automated game-theoretic security reasoning," in *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*, 2023, pp. 1407–1421.

[25] 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.

[26] 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*, 2020, pp. 601–619.

[27] M. Carlsten, H. Kalodner, S. M. Weinberg, and A. Narayanan, "On the instability of bitcoin without the block reward," in *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, 2016, pp. 154–167.

[28] T. Börgers, *An introduction to the theory of mechanism design*. Oxford university press, 2015.

[29] J. D. Hartline and T. Roughgarden, "Simple versus optimal mechanisms," in *Proceedings of the 10th ACM Conference on Electronic Commerce (EC '09)*. ACM, 2009, pp. 225–234.

[30] D. Bergemann and J. Välimäki, "The dynamic pivot mechanism," *Econometrica*, vol. 78, no. 2, pp. 771–789, 2010.

[31] R. B. Myerson, "Optimal auction design," *Mathematics of operations research*, vol. 6, no. 1, pp. 58–73, 1981.

[32] L. Donno, "Optimistic and validity rollups: Analysis and comparison between optimism and starknet," *arXiv preprint arXiv:2210.16610*, 2022.

[33] W. Ou, S. Huang, J. Zheng, Q. Zhang, G. Zeng, and W. Han, "An overview on cross-chain: Mechanism, platforms, challenges and advances," *Computer Networks*, vol. 218, p. 109378, 2022.

[34] J. Poon and T. Dryja, "The bitcoin lightning network: Scalable off-chain instant payments," 2016.

[35] M. Xu, Y. Zhang, and S. Zhong, "Silentower: A robust, scalable and secure watchtower with silent executors," in *2023 42nd International Symposium on Reliable Distributed Systems (SRDS)*. IEEE, 2023, pp. 173–186.

[36] E. Jaghjough Lamrani, "The staking mechanisms in ethereum," 2024.

[37] Z. He and J. Li, "Contract enforcement and decentralized consensus: The case of slashing," *Available at SSRN 4036000*, 2022.

[38] P. NFT, "Pow nft: The first ever mineable nft," https://www.pownft.com/, 2025, accessed, October 2025.

[39] S. Eskandari, M. Salehi, W. C. Gu, and J. Clark, "Sok: Oracles from the ground truth to market manipulation," in *Proceedings of the 3rd ACM Conference on Advances in Financial Technologies*, 2021, pp. 127–141.

[40] L. Breidenbach, C. Cachin, B. Chan, A. Coventry, S. Ellis, A. Juels, F. Koushanfar, A. Miller, B. Magauran, D. Moroz *et al.*, "Chainlink 2.0: Next steps in the evolution of decentralized oracle networks," *Chainlink Labs*, vol. 1, pp. 1–136, 2021.

[41] J. Adler, R. Berryhill, A. Veneris, Z. Poulos, N. Veira, and A. Kastania, "Astraea: A decentralized blockchain oracle," in *2018 IEEE international conference on internet of things (IThings) and IEEE green computing and communications (GreenCom) and IEEE cyber, physical and social computing (CPSCom) and IEEE smart data (SmartData)*. IEEE, 2018, pp. 1145–1152.

[42] F. Zhang, D. Maram, H. Malvai, S. Goldfeder, and A. Juels, "Deco: Liberating web data using decentralized oracles for tls," in *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, 2020, pp. 1919–1938.

[43] K. Kursawe, "Wendy, the good little fairness widget: Achieving order fairness for blockchains," in *Proceedings of the 2nd ACM Conference on Advances in Financial Technologies*, 2020, pp. 25–36.

[44] A. Orda and O. Rottenstreich, "Enforcing fairness in blockchain transaction ordering," *Peer-to-peer Networking and Applications*, vol. 14, no. 6, pp. 3660–3673, 2021.

[45] M. Kelkar, F. Zhang, S. Goldfeder, and A. Juels, "Order-fairness for byzantine consensus," in *Annual International Cryptology Conference*. Springer, 2020, pp. 451–480.

[46] M. Kelkar, S. Deb, and S. Kannan, "Order-fair consensus in the permissionless setting," in *Proceedings of the 9th ACM on ASIA Public-Key Cryptography Workshop*, 2022, pp. 3–14.

[47] C. Cachin, J. Mićić, N. Steinhauer, and L. Zanolini, "Quick order fairness," in *International Conference on Financial Cryptography and Data Security*. Springer, 2022, pp. 316–333.

[48] ZKFair, "Decentralized prover network," https://docs.zkfair.io/decentralized-prover-network, 2024, accessed: 2024-05-17.

[49] A. Engineering, "Proof of stake vs proof of work: A guide to consensus mechanisms," https://aleo.org/post/proof-of-stake-vs-proof-of-work/, 2025, published: 2025-04-11. Accessed: 2025-11-06.

[50] Z. Team, "Zkfair documentation," https://docs.zkfair.io/, 2024, accessed: 2024-05-20.

[51] S. Das, V. Krishnan, I. M. Isaac, and L. Ren, "Spurt: Scalable distributed randomness beacon with transparent setup," in *2022 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2022, pp. 2502–2517.

[52] A. Bhat, N. Shrestha, Z. Luo, A. Kate, and K. Nayak, "Randpiper–reconfiguration-friendly random beacons with quadratic communication," in *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, 2021, pp. 3502–3524.

[53] H. Chung and E. Shi, "Foundations of transaction fee mechanism design," in *Proceedings of the 2023 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. SIAM, 2023, pp. 3856–3899.

[54] S. Industries, "Starknet: A decentralized validity-rollup (zk-rollup)," https://starkware.co/starknet/, 2023, accessed: 2024-05-17.

[55] Polygon Technology, "Polygon," https://polygon.technology/, 2023, accessed: 2024-05-17.

[56] Z. Li, J. Li, Z. He, X. Luo, T. Wang, X. Ni, W. Yang, X. Chen, and T. Chen, "Demystifying defi mev activities in flashbots bundle," in *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*, 2023, pp. 165–179.

[57] S. Micali, M. Rabin, and S. Vadhan, "Verifiable random functions," in *40th annual symposium on foundations of computer science (cat. No. 99CB37039)*. IEEE, 1999, pp. 120–130.

[58] T. Todo, A. Iwasaki, and M. Yokoo, "False-name-proof mechanism design without money," in *The 10th International Conference on Autonomous Agents and Multiagent Systems-Volume 2*, 2011, pp. 651–658.

[59] J.-J. Laffont and D. Martimort, "Mechanism design with collusion and correlation," *Econometrica*, vol. 68, no. 2, pp. 309–342, 2000.

[60] T. Roughgarden, "Transaction fee mechanism design," *ACM SIGecom Exchanges*, vol. 19, no. 1, pp. 52–55, 2021.

[61] A. Kiayias, A. Russell, B. David, and R. Oliynykov, "Ouroboros: A provably secure proof-of-stake blockchain protocol," in *Annual international cryptology conference*. Springer, 2017, pp. 357–388.

[62] P. Daian, R. Pass, and E. Shi, "Snow white: Robustly reconfigurable consensus and applications to provably secure proof of stake," in *International conference on financial cryptography and data security*. Springer, 2019, pp. 23–41.

[63] V. Buterin *et al.*, "Ethereum white paper," *GitHub repository*, vol. 1, no. 22-23, pp. 5–7, 2013.

[64] H. A. David and H. N. Nagaraja, *Order statistics*. John Wiley & Sons, 2004.

Appendix A. Practical Considerations of Prrr

First, some blockchain protocols, such as Ethereum [63], require a minimum fee for each transaction. This minimum fee is not awarded to validators but is instead *burned*, i.e., permanently removed from circulation. The Prrr protocol can simply add an amount equal to the minimum fee to the reward paid to the publishers of the (up to) two included reports. Thus, the minimum fee would not affect our analysis. Since the protocol includes at most two reports per block, the additional cost is nominal.

Second, most blockchain protocols process transactions within a block sequentially. When executing a transaction, the system cannot access information about subsequent transactions in the same block. As a result, when processing a transaction containing a report, the smart contract cannot immediately determine whether it is the last report in the block and allocate the reward accordingly. To address this, the smart contract processes the first report as usual and records any additional reports for later evaluation. The final reward calculation can then be performed after all relevant information is available.

Third, in practical systems, publishers can specify a validity duration window for each transaction, eliminating the need to publish transactions at every step. If a publisher publishes a transaction with a duration window spanning multiple steps, we can simply treat it as if the transaction is published for each step within that window. Then, it does not affect the analysis of our protocol. In our protocol, publishers can publish all report transactions by specifying a transaction window that encompasses all steps within the report publication window, achieving the same outcome.

Appendix B. Analysis of Random-Value Functions

We analyze the logarithmic (§B.1) and polarized (§B.2) random-value functions separately. For each case, we first discuss parameter selection to ensure Reward Monotonicity (Property 1) and Skipping Resistance (Property 2). Then, we analyze the expected cost incurred by the smart contract when using the random-value function.

B.1. The Logarithmic Random-Value Function

Recall that $r_{\min}$ is the minimum of the random-value function, $H(\cdot)$ is the random oracle, $\lambda > 0$ is a parameter that controls the distribution of rewards. Given a report *Rpt* and a random string S, recall that the logarithmic random-value function is (§5.5):

$$\mathit{RV}_{\mathit{log}}(\mathit{Rpt}, S) = r_{\min} - \frac{1}{\lambda}\ln(1 - H(\mathit{Rpt}||S)) \ .$$

Then the distribution of $\mathit{RV}_{\mathit{log}}(\mathit{Rpt}, S) - r_{\min}$ follows an exponential distribution with parameter λ:

$$\Pr(\mathit{RV}_{\mathit{log}}(\mathit{Rpt}, S) - r_{\min} \geq r) = \Pr\left(-\frac{1}{\lambda}\ln(1 - H(\mathit{Rpt}||S)) \geq r\right) = e^{-\lambda r}$$

For any N reports $\{\mathit{Rpt}_i\}_{i=1}^{N}$ and any random string S, the random variables $\mathit{RV}_{\mathit{log}}(\mathit{Rpt}_i, S) - r_{\min}$ are i.i.d. exponential random variables with parameter λ. By the memoryless property of the exponential distribution, the difference between the highest and second-highest values also follows an exponential distribution with parameter λ [64]. Therefore, the expected total reward for all publishers when there are N reports is fixed to $\mathit{RAllPub}(N) = \frac{1}{\lambda}$, regardless of N. Thus, Reward Monotonicity (Property 1) is satisfied trivially. To satisfy Skipping Resistance (Property 2), we need $\mathit{RAllPub}(N_{\max}) = \frac{1}{\lambda} < r_{\min}$. Therefore, by setting $\lambda > \frac{1}{r_{\min}}$, we ensure that both properties are satisfied.

The expected reward for validators is the expected second-highest value among N reports and the expected reward for publishers is the expected difference between the highest and second-highest value among N reports. Therefore, the total expected cost for the smart contract is the expected highest value among N reports.

Denote by $H_N = \sum_{i=1}^{N} \frac{1}{i}$ the N-th harmonic number. For N i.i.d. random variables following the exponential distribution with parameter λ, the expected highest value among them is $\frac{H_N}{\lambda}$ [64]. Therefore, for the logarithmic random-value function, the total expected cost for the smart contract when there are N reports is $r_{\min} + \frac{H_N}{\lambda}$. Since H_N is $O(\log N)$, the total expected cost for the smart contract increases logarithmically with the number of reports.

B.2. The Polarized Random-Value Function

Recall that $r_{\min}$ and $r_{\max}$ are the two possible outcomes of the polarized random-value function, where $r_{\max} > r_{\min}$, p is the probability of receiving $r_{\max}$, and $H(\cdot)$ is the random oracle. Given a report *Rpt* and a random string S, recall that the polarized random-value function is (§5.5):

$$\mathit{RV}_{\mathit{polarized}}(\mathit{Rpt}, S) = \begin{cases} r_{\max} & \text{if } H(\mathit{Rpt}||S) \leq p \\ r_{\min} & \text{otherwise} \end{cases}$$

Publishers have positive revenue only when there is exactly one report that has the high value $r_{\max}$. When there are N reports in total, the probability is $N \times p \times (1-p)^{N-1}$. Therefore, the expected total reward for all publishers is

$$\mathit{RAllPub}(N) = N \times p(1-p)^{N-1}(r_{\max} - r_{\min}) \ . \quad (33)$$

For monotonicity, since $\frac{\mathit{RAllPub}(N)}{\mathit{RAllPub}(N-1)} = \frac{N}{N-1} \times (1-p)$, $\mathit{RAllPub}(N)$ is monotonically increasing with N for $N \leq \frac{1}{p}$. Therefore, by setting $p \leq \frac{1}{N_{\max}}$, the monotonicity property is satisfied.

In addition, to satisfy Skipping Resistance (Property 2), we need $\mathit{RAllPub}(N_{\max}) < r_{\min}$ for all $N \leq N_{\max}$. Due to

monotonicity, it suffices to ensure $\mathit{RAllPub}(N_{\max}) < r_{\min}$. The condition becomes:

$$r_{\max} < (1 + \frac{1}{N_{\max} p (1-p)^{N_{\max}-1}}) r_{\min} \ .$$

In all, by setting $p \leq \frac{1}{N_{\max}}$ and $r_{\max} < (1 + \frac{1}{N_{\max} p (1-p)^{N_{\max}-1}}) r_{\min}$, both properties are satisfied for the polarized random-value function.

Recall that the expected total cost for the smart contract is the expected highest value among N reports (§B.1). For the polarized random-value function, the maximal expected cost to the smart contract is $r_{\max}$. The expected cost to the smart contract when there are N reports is the probability that at least one report has the high value $r_{\max}$ times $r_{\max}$ plus the probability that no report has the high value $r_{\max}$ times $r_{\min}$:

$$\begin{aligned} & E_S[\max(\max_{1 \leq i \leq N} \mathit{RV}_{\mathit{polarized}}(\mathit{Rpt}_i, S), r_{\min})] \\ =& (1 - (1-p)^N) \times r_{\max} + (1-p)^N \times r_{\min} \\ =& r_{\min} + (1 - (1-p)^N) \times (r_{\max} - r_{\min}) \end{aligned}$$

Appendix C. Validator Deviations

We analyze validator deviations when all publishers follow the Prrr protocol. When multiple reports share the highest value, the validator has several best responses: she may place any two of these highest-value reports as the first and second reports. All such choices are consistent with the Prrr protocol and yield identical publisher revenue, since the reward, which is the difference between the highest and second-highest values, is zero. If the highest value is unique but several reports share the second-highest value, the validator may select any two of these second-highest reports as the first and second included reports. This does not affect the validator's revenue, but it strictly reduces the publisher's revenue for the highest-value report to zero. In all other cases, the validator's best response is unique: she can only obtain the second-highest value reward by following the Prrr protocol (Equation 14), and any deviation would strictly decrease her revenue.

Observation 5. *In the game $\mathcal{G}$, suppose all publishers follow the Prrr protocol. For a validator V^k, if the highest value among all published reports is unique and the second-highest value is also unique, then any deviation from the Prrr protocol strictly decreases the validator's revenue.*

For the logarithmic value function, the probability that two reports have exactly the same value is zero since it is a continuous distribution. Therefore, we have

Corollary 2. *In the game $\mathcal{G}(\mathit{RV}_{\mathit{log}})$ where $\mathit{RV}_{\mathit{log}}$ is the logarithmic random-value function, suppose all publishers follow the Prrr protocol. Then, with probability 1, any deviation from the Prrr protocol strictly decreases the validator's revenue.*

For the polarized value function, since there are only two possible reward values, if there are at least three reports published, the validator always has multiple best responses when all publishers follow the Prrr protocol.