Rationally Analyzing SHELBY: Proving Incentive Compatibility in a Decentralized Storage Network

Michael Crystal[1], Guy Goren[2], and Scott Duke Kominers[3]

[1] Stanford University
[2] Aptos Labs
[3] Harvard University and a16z crypto

Abstract. Decentralized storage is one of the most natural applications built on blockchains and a central component of the Web3 ecosystem. Yet despite a decade of active development—from IPFS and Filecoin to more recent entrants—most of these storage protocols have received limited formal analysis of their incentive properties. Claims of incentive compatibility are sometimes made, but rarely proven. This gap matters: without well-designed incentives, a system may distribute storage but fail to truly decentralize it.

We analyze SHELBY—a storage network protocol recently proposed by Aptos Labs and Jump Crypto—and provide the first formal proof of its incentive properties. Our game-theoretic model shows that while off-chain audits peer alone collapse to universal shirking, SHELBY's combination of off-chain audits with occasional on-chain verification yields incentive compatibility under natural parameter settings. We also outline a simple modification that strengthens the protocol's collusion-resilience.

1 Introduction

One of the longest-running and arguably most natural applications on blockchains is the implementation of decentralized storage networks. Beginning in the 2010s with IPFS [7], Sia [35], Filecoin [26], Storj [33], and Arweave [37], these systems pioneered blockchain-based approaches to content addressing and incentivized storage. More recently, the ecosystem has expanded with networks such as Crust [10], Celestia [1], EigenDA [16] and Walrus [13], among others. These systems make it possible to leverage a dispersed and flexibly extensible network of service providers to store data, rewarding them for doing so. However, there remains a lack of formal incentive analysis for most of these protocols.

While *distributing* storage across many nodes is an important systems property, *decentralizing* storage—in the sense of aligning incentives so that independent participants reliably store their assigned data—remains a critical challenge. This distinction is often blurred: much of the existing analysis of storage protocols remains at the distributed-systems level, proving safety and liveness under the assumption that some majority of nodes are "naively honest" (for a recent

survey, see [21]). Such assumptions may suffice for fault tolerance, but they fail to address why rational, self-interested actors would follow the protocol in the first place. The economic issue is straightforward: distributed but centrally controlled services such as AWS or Google Cloud internalize both the gains and losses from reliable storage, whereas in a decentralized network, individual storage providers do not fully internalize these externalities—bearing at most a fraction of the system-wide consequences if they choose to shirk. Audit mechanisms aim to address this by penalizing misbehavior, but auditing every action on-chain is prohibitively expensive. To keep costs down, protocols rely heavily on off-chain audits, but this creates a second incentive challenge: auditors themselves must be incentivized to perform faithful audits and to report them truthfully. This "who audits the auditors" problem makes clear that without explicitly incentive-compatible design, decentralized storage risks collapsing into little more than distributed storage run on wishful assumptions.

Recently, Aptos Labs and Jump Crypto jointly introduced Shelby [19,31], a decentralized storage protocol with a two-tiered auditing system. In Shelby, storage providers audit one another off-chain, and a subset of those audits are occasionally checked through a more costly but perfectly-verifiable on-chain review process. This design directly addresses the "who audits the auditors" problem described above, and makes Shelby a natural case study for formal analysis. Unlike with most existing storage protocols, which provide little more than distributed-systems–style reasoning, we are able to provide a game-theoretic characterization of Shelby's incentive compatibility.

In our model, a network of storage providers (SPs) make independent decisions about whether to store their assigned data, whether to exert effort to audit others' storage behavior, and whether to truthfully report the results of any audits they conduct. We formalize the challenge of "auditing the auditors," showing that when auditors' reports must simply be relied on, with no backstop to enforce truthful auditing, the unique pure-strategy Nash equilibrium is one in which no agent provides storage or auditing services, and yet all agents (falsely) assert that all audits were conducted and passed successfully. Following the Shelby design, we then introduce the possibility of occasional, costly on-chain verification of SPs' off-chain audits, and show that this reverses the conclusion: so long as protocol rewards and penalties (slashing) are calibrated correctly, occasional on-chain audits are sufficient to render the entire protocol incentive compatible. We furthermore demonstrate a sense in which these positive incentives are robust to collusion among groups of SPs. Finally, we use our framework to show that requiring SPs to submit vector commitments over audit data (in addition to their audit reports) would strengthen Shelby's collusion-resilience.

The key economic insight underlying our results is that infrequent perfectly-verifiable audits are sufficient to enforce truthful auditing off-chain; this means that failures of storage provision will be detected and properly penalized, which makes the storage process incentive compatible as well. This mechanism—using costly ground-truth checks to align incentives around lower-cost interim audits—may apply more broadly in other decentralized infrastructure contexts.

1.1 Our Contributions

In summary, our contributions are fourfold:

- We introduce a non-cooperative game theory model of service provision in a decentralized storage network with cross-SP auditing.
- We use this model to formalize the "auditing the auditors" problem, a general incentive challenge that arises in the autarky equilibria of these networks.
- We then provide the first formal incentive analysis of the Shelby protocol, showing that the two-tiered auditing structure Shelby uses fully addresses the aforementioned challenge, making it possible to achieve full incentive compatibility of both storage provision and auditing when SPs operate independently. Moreover, we show that the truthful equilibrium is robust to coalitions of up to half the SP population when coalition members do not have a way to commit to support each other in the auditing phase.
- Via our framework, we identify a potential improvement to the Shelby protocol that would further reinforce its collusion-resilience: if SPs are additionally required to submit vector commitments to audit responses alongside their audit reports, the robustness to collusion carries over fully even when coalition members have commitment power.

1.2 Related Work

To our knowledge, our work is the first to provide a game-theoretic foundation for full incentive compatibility of a decentralized storage protocol in production.

Many have highlighted the importance of participation incentives—in the form of payments and other direct rewards—in decentralized storage networks (see, e.g., [14,38]). However, these analyses have typically been limited to high-level questions such as the need for fairness in reward distribution and ensuring that service providers can cover their operational costs, rather than reasoning about how to make the overall system fully incentive-aligned.

Meanwhile, most public analyses of storage protocols emphasize distributed-systems properties (such as safety, liveness, and fault tolerance), stopping short of game-theoretic incentive analysis. For example, the Filecoin white paper [26] states a desired incentive-compatibility property but does not provide a formal proof; recent work establishes fault-tolerance guarantees [36], but, to our knowledge, no publicly-available formal incentive compatibility argument for Filecoin exists. Walrus [13] presents fault-tolerance guarantees without game-theoretic claims about network incentives. Arweave [37] informally argues that its proof-of-access leads to a dominant strategy and that P2P rules incentivize "pro-social actions," but does not furnish a formal proof. Storj [33] discusses "incentive alignment" via audits, escrow, and reputation without formal incentive compatibility theorems. Celestia [1,2] provides rigorous data-availability and sampling guarantees, but—as with the others—does not present a formal proof of incentive compatibility. Sia [35] and EigenDA [16] describe mechanisms (e.g., contracts, slashing, proof-of-custody) that govern incentives, yet do not provide formal

game-theoretic proofs. Likewise, Crust's economy white paper [11] articulates economic design claims rather than formal incentive compatibility results.

A notable example from outside the storage domain is Chainlink [8], which introduces a two-tier watchdog mechanism for oracle verification, pairing cheap first-level checks with costlier fallback validation. While Chainlink's architecture shares structural resonance with SHELBY's two-tier audit system, their work focuses on unverifiable off-chain data—a substantial difference because SHELBY's guarantees directly key off of the possibility of on-chain verification. Moreover, Chainlink [8] does not provide formal game-theoretic guarantees.

Meanwhile, other studies of enforcement through auditing mechanisms in distributed storage have focused on security and computational challenges (see, e.g., [15] and the references therein), rather than incentive-alignment of auditing.

The closest work to ours that we are aware of is the recent paper of Vakilinia et al. [34], which proposes a dominant-strategy incentive-compatible mechanism for decentralized storage under the assumption of a trusted (off-chain) auditing oracle network. Like SHELBY, the protocol of [34] leverages infrequent costly verification to provide proper incentives; however, [34] uses those infrequent audits to confirm proper storage, whereas SHELBY uses them to confirm auditors' reports. As we show in our game-theoretic analysis, pushing the verification step to the auditing level removes the need to have a trusted storage oracle network—and because this drastically lowers the overall verification costs, it enables SHELBY to bring the backstop verification step on-chain.[4]

All of that said, there is significant game-theoretic precedent in the analysis of peer-to-peer storage and information networks (see [20] for a recent survey). And there has been recent research on market designs for ensuring truthful information revelation in decentralized physical infrastructure networks [25].

And a number of authors have considered ways of compounding scarce audits into broader system-wide incentives (see, e.g., [22,17,12]). Foundational to all of this research are the classic Alchian and Demsetz [3] framework that formalized the monitoring problem in team production, as well as the more recent work of Rahman [27], which introduced a game-theoretic incentive scheme for "monitoring the monitor."

2 The SHELBY Protocol

We briefly summarize the design of the SHELBY protocol (for further detail, see [19]) before proceeding to our formal analysis in Section 3. First, we describe SHELBY's system mechanics (Section 2.1), along with the associated payments for service provision and penalties for misbehavior (Section 2.2). Then, we summarize the system's usage of erasure coding for fault-tolerance (Section 2.3).

[4] Another differentiator is that whereas [34] relies on the client detecting potential storage failures and requesting audits, SHELBY's randomized auditing scheme removes the need for this. Indeed, as our analysis in the sequel demonstrates, SHELBY obtains incentive compatibility without any direct appeal or reference to the client.

2.1 System Mechanics

Shelby delegates coordination and verification to a trusted blockchain, while leaving the bulk of communication and auditing off-chain.

Data Storage. Each storage provider (SP) stores a set of data chunks and maintains an on-chain vector commitment representing the current state of its stored data. When a chunk is added or modified, the SP updates their commitment. Currently, Shelby implements vector commitments [9] using Merkle trees [23]: each chunk is a leaf in the Merkle structure, and the commitment is the Merkle root. Chunks are $\sim$1 MiB in size, and corresponding inclusion proofs are $\sim$1 KiB.

Off-Chain Audits. Public randomness, produced on-chain by Aptos [5], serves as a shared seed visible to all SPs. At each block, this randomness is used to select chunks for auditing, with each selected chunk associated with a unique, commonly known, SP (the auditee). That SP must then provide a Merkle inclusion proof for the selected chunk to a subset of other SPs designated as auditors.[5] This audit process continues over the course of an audit epoch. By the end of the epoch, each SP has acted both as an auditee and as an auditor multiple times.

In its auditor role, an SP collects the responses it received from auditees into an N-element array, in which element i is a binary vector indicating the outcomes of the audits of SP i. A value of 1 represents a successful response; a value of 0 indicates a failure or a missing response. If we let f_{st} denote the expected number of audits that each SP undergoes per epoch in their role as an auditee, then each audit report contains approximately $N \cdot f_{\text{st}}$ entries. Auditors retain the received inclusion proofs in local storage until the end of the *next* epoch, as they may be required to present them for inspection during that epoch in the on-chain verification process we describe below.

Audit Report Aggregation. At the end of each audit epoch, each SP (in its role as an auditor) submits its audit report on-chain. Based on the full set of submitted reports, an audit score is computed for each auditee. Specifically, for each audit of SP i, the audit is counted as passing if a strict majority (more than 50%) of i's auditors reported a 1. Non-votes (i.e., missing responses) are interpreted as 0s; and if an SP fails to submit its audit report, it is treated as if it had submitted a report consisting entirely of 0s. The resulting *audit score* of SP i, denoted $\gamma_i \in [0, 1]$, is defined as the share of *passing* audits, after voting, out of all the audits i was subject to during the epoch. This computation is performed on-chain and is transparent to all participants.

On-Chain Audit Report Inspection. In addition, each auditor's submitted report undergoes random inspection. Using fresh on-chain randomness, each 1 entry from the report is randomly selected for inspection with probability p_{au}. The

[5] In practice, Shelby uses selected subcommittees of auditors to audit a given auditee—but for expositional simplicity in our discussion and analysis, we assume that each auditee sends proofs to all other SPs.

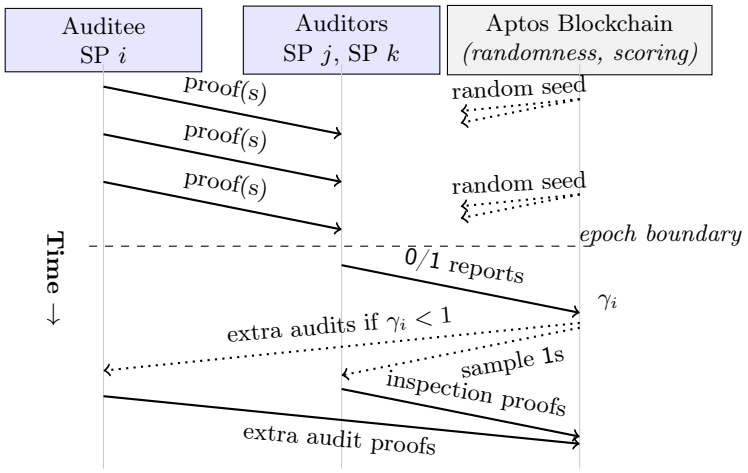

Fig. 1: Sequence of events in an audit epoch: SP i is audited by a group of SPs who submit reports to the blockchain. The blockchain computes audit scores, inspects auditors, and may assign extra audits if needed.

auditor is then required to provide the inclusion proof corresponding to that entry for on-chain verification. This cryptographic proof is verified directly from on-chain data (specifically, the vector commitment) and is not subject to majority voting of any kind. Failure to provide a valid proof in a timely manner is considered a failed inspection.

Finally, any SP i with an audit score $\gamma_i < 1$ is subjected to additional random audits performed directly on-chain. Let $\alpha_i = 1 - {\gamma_i}^2$. Then, $\alpha_i \cdot C_{\max}$ of i's chunks are randomly selected for direct on-chain inclusion proof verification, where $C_{\max}$ is a system parameter establishing the maximum audit volume. While failure in these additional on-chain audits results in slashing, successfully passing them does not restore the reduced payments associated with a low score. This asymmetry is intentional: it prevents SPs from relying on low-frequency, on-the-fly reconstruction to pass on-chain audits while neglecting their obligations under the more frequent peer audits.[6] The full process flow is shown in Figure 1.

2.2 Payments and Penalties

Each SP receives two types of participation rewards at the end of each epoch—one for storing its assigned data, and one for (properly) performing audits:

[6] Notice that a coalition could in principle prevent an SP from receiving its storage reward by colluding to report 0 even when that SP has passed all audits successfully; however, if the coalition does so, then its members forgo all audit rewards, since those are received only for successful audits. This could not be rational without the coalition receiving extra-market utility—such as directly valuing disrupting the network—which we do not consider here (see Footnote 12).

- **Storage rewards.** SP i's storage rewards are proportional to the volume of data i stores, measured in chunks, scaled by i's audit score, and are granted only when its audits count as passing. An audit counts as passing if a strict majority of SP i's auditors reports a 1. Thus, SP i receives $\gamma_i \cdot r_{\text{st}} \cdot \mathbb{1}_{\left|\{j:\rho_j^i=1\}\right|>N/2}$ per stored chunk per epoch, where r_{st} is a fixed per-chunk storage reward and $\rho_j^i \in \{0,1\}$ denotes SP j's audit report on i.
- **Auditor rewards.** For each audit where SP i serves as an auditor, it receives a fixed reward r_{au} if it reports the audit as successful (i.e., submits a 1 entry in the audit report).[7]

Misbehaving SPs may also become susceptible to slashing. All slashing events are carried out on-chain and are forensics-based. In particular, slashing cannot be imposed through voting, thus avoiding a scenario where a majority can slash a correctly-behaving minority. Slashing may occur in the following two cases:

- **On-chain audit-the-auditor inspection.** An auditor that fails to provide a valid inclusion proof during random inspection is slashed by S_{au}.
- **Additional on-chain storage audits.** An SP with a low audit score that is selected for direct on-chain audits and fails to provide a valid inclusion proof is slashed by S_{st}.

2.3 Erasure Coding and Data Reconstruction

The Shelby storage system employs erasure coding [29] to achieve efficient fault-tolerance. Very roughly, a user file is divided into multiple data chunks, and additional redundant chunks are generated (see [28] for a canonical construction); the result is a set of chunks, each containing partial information about the original file, such that the full file can be reconstructed from any k chunks. For simplicity, we conflate this k-out-of-N threshold with the requirement that at least two-thirds of SPs serve their assigned chunks faithfully. The key advantage of erasure coding is that it enables redundancy with substantially lower storage overhead than full replication. Under the two-thirds threshold assumption, the overhead is approximately 50%, compared to at least 3× for full replication.[8]

This redundancy has implications for the auditing scheme. In particular, an auditee that fails to store a chunk might attempt to reconstruct it on-the-fly during an audit. However, since each chunk is distinct, reconstruction is not as simple as requesting a copy from a single peer. Instead, the auditee must retrieve data from multiple different chunks—held by different SPs—to recreate its own. Shelby's encoding and reconstruction scheme requires information from at least k distinct chunks to reconstruct a single chunk. Given that data in the system is read at the granularity of full chunks,[9] this implies that on-the-fly

[7] This reward does not depend on the amount of data stored by i, and is designed to encourage active participation in the protocol.

[8] This approach is standard in modern large-scale storage systems [32].

[9] In specific cases (e.g., recovery of tens of GiBs at an SP), the system supports data formats allowing non-complete chunk access. This is primarily used for crash recovery and involves high data volumes, making it impractical for on-the-fly audits.

reconstruction requires the auditee to read at least k chunks from k different SPs.

Denote by c_{r} the cost of reading a single chunk from another SP. Then, answering an audit by reconstructing the chunk on-the-fly incurs a cost of at least $k \cdot c_{\mathrm{r}}$. By contrast, the expected cost of storing the chunk locally for the duration of an epoch is c_{st}, during which it is audited f_{st} times in expectation at a per-audit cost of c_{au}.[10] This yields the following:

Observation 1 *Whenever*

$$f_{\mathrm{st}} \cdot k \cdot c_{\mathrm{r}} > c_{\mathrm{st}}, \tag{1}$$

answering audits by storing the associated chunks strictly dominates answering them via on-the-fly reconstruction.

Based on SHELBY's documentation [19,31] and pricing data from real-world systems [4,18,24], we have $k = 10$, $f_{\mathrm{st}} > 3$, and estimate $c_{\mathrm{st}}/c_{\mathrm{r}} \approx 2.5$ (where f_{st} and c_{st} are normalized to a monthly basis). Thus, in practice, we expect that (1) holds, so storing chunks dominates reconstruction in audit response.

Table 1 summarizes the SHELBY protocol parameters relevant to our analysis.

3 Incentive Analysis

Now, we introduce our formal framework for analyzing the storage and auditing incentives under the SHELBY protocol. Building on the notation and terminology introduced in Section 2, we consider a population I of *storage providers* (*SPs*) indexed by $i = 1, 2, \ldots, N$. Each SP's action space consists of three components:

1. a binary decision as to whether to store the assigned data;
2. a vector of binary choices as to whether to perform assigned audits; and
3. a vector of audit scores assessing the compliance of other SPs.

[10] Note that we abstract away from the costs of proof generation and validation. Although precise costs depend on implementation details, a representative deployment recommended by [30] involves each SP precomputing a Merkle tree per chunk and storing it alongside the chunk. This adds negligible overhead: approximately 32 KiB atop the 1 MiB chunk. During an audit, the auditee retrieves the precomputed Merkle inclusion proof and sends it to the auditor, who verifies it by performing ∼10 SHA-256 hashes. The auditor temporarily stores the proof for one additional epoch to support possible audit-the-auditor queries. Thus, c_{st} (the auditee's per-chunk cost per epoch) is dominated by the storage cost of 1 MiB for one month. The cost of computing the full Merkle tree (∼2048 hashes) is amortized and negligible relative to storage, as computing 2000 hashes is ∼3–4 orders of magnitude cheaper than storing 1 MiB for a month. Conversely, c_{au} (the auditor's per-audit cost) is dominated by the networking costs of fetching the proof, while the runtime cost of computing ∼11 hashes during verification is much lower; storing the ∼1.3 KiB proof for a few hours costs roughly two orders of magnitude less than the hash computations.

Parameter	Description
c_{st}	Cost to store a chunk per epoch.
c_{au}	Cost of auditing a storage provider.
c_{r}	Cost of reading a chunk from another storage provider.
r_{st}	Storage reward per epoch.
r_{au}	Auditor reward per successful audit.
f_{st}	Frequency that a stored chunk is audited per epoch.
p_{au}	Probability that a 1 entry in an auditor report is selected for on-chain proof verification.
S_{st}	Slashing penalty for on-chain storage audit failure.
S_{au}	Slashing penalty for failing audit-the-auditor verification.

Table 1: Shelby Protocol Parameters

Formally, we let $\sigma_i = (s_i, e_i, \rho_i) \in \{0,1\} \times \{0,1\}^{N-1} \times \{0,1\}^{N-1}$ denote SP i's action space, where $s_i = 1$ indicates that SP i stores its assigned data; $e_i^j = 1$ indicates that SP i properly conducts their assigned audit of SP j; and $\rho_i^j = 1$ indicates that SP i reports SP j as having stored its assigned data. Note that we implicitly subsume the possibility that a storage provider may choose to store its assigned data yet nevertheless refuse to provide a storage proof to an auditor; in this case, the audit is "failed" from the perspective of the auditor, and so the auditor reports 0.[11] The (gross) utility of SP i is given by

$$u_i(\sigma_i, \sigma_{-i}) = \gamma_i \cdot r_{\text{st}} \cdot \mathbb{1}_{\left|\{j:\rho_j^i=1\}\right| > N/2} + r_{\text{au}} \sum_{j \neq i} \rho_i^j - s_i \cdot c_{\text{st}} - c_{\text{au}} \cdot \sum_{j \neq i} e_i^j, \quad (2)$$

where r_{st} and r_{au}, respectively, denote the rewards for being reported as compliant and for submitting audit scores that are 1s; while c_{st} and c_{au} are, respectively, the cost of storing data and the cost of conducting an individual audit.[12] For simplicity and ease of notation, we express the storage reward payoff on a per-chunk basis; this is without loss of generality, as extending it to total stored data would not affect the analysis. We interpret r_{au} as the net reward for submitting a report with value 1, since submitting an on-chain report incurs a cost regardless of whether an audit was actually performed. A storage provider who opts not to store the data (i.e., who sets $s_i = 0$) may still attempt to reconstruct the data from other providers at a reconstruction cost of c_{r}.[13]

[11] Note that an SP may choose not to conduct audits ($e_i^j = 0$) while still submitting a vector of audit scores. While the space of possible audit reports is richer in practice, the system treats any report that is not exactly 1 as equivalent to 0. As a result, without loss, we can model each audit report as a binary variable in our analysis.

[12] Note that we implicitly rule out extra-market sources of utility such as agents benefiting from shutting down or otherwise disrupting the overall network.

[13] In practice, a storage provider attempting to retrieve data they did not store risks failure to do so. However, we take a conservative modeling approach and assume that such retrieval is always successful.

Additionally, for notational convenience, we set $f_{\text{st}} = 1$.[14]

3.1 The "Who-Will-Audit-The-Auditor" Problem

As a benchmark, we analyze SHELBY's off-chain SP audits as an isolated game. Formalizing the "auditing-the-auditors" problem, we show that if the audit mechanism relies solely on these off-chain audits, then the fully dishonest strategy is the unique equilibrium.

Formally, we say that SP i follows a *fully dishonest* strategy if they choose

$$\sigma_i^\circ(s_i, e_i, \rho_i) = (0,\, 0, \ldots, 0,\, 1, \ldots, 1);$$

i.e., a fully dishonest SP does not store data, assigns 1 audit scores to all others, and refrains from conducting audits.

Proposition 1. *With only off-chain audits, the unique pure strategy Nash equilibrium is (mutual) full dishonesty, i.e., for all $i \in I$, $\sigma_i = \sigma_i^\circ$.*

The main intuition behind Proposition 1 is the problem of "who audits the auditor?" A mechanism that relies solely on off-chain audits by SPs offers no guarantee that audits are conducted truthfully. Without a way to verify or enforce audits, SPs have a dominant strategy to avoid audit costs and simply report successful outcomes (1s).

In a fully dishonest equilibrium, SPs receive all rewards without incurring costs. This outcome is robust to collusion, so long as no external funds enter the system. Thus, we see that off-chain auditing alone is insufficient—aligning incentives for storage requires a way of aligning incentives around audits.

3.2 Truthful Behavior under On-Chain Auditing

In this section, we augment the off-chain auditing scheme with a provably accurate on-chain mechanism, which targets SPs suspected of underperformance and at the same time verifies a subset of audits to ensure correct reporting. Let p_{au} denote the probability that an auditor who reports a value of 1 is selected for on-chain verification of their audit proof, and let $S_{\text{au}} \geq 0$ represent the slashing penalty for failing this verification. SPs whose audits fail on-chain verification also lose the associated audit rewards r_{au}.[15]

We begin by identifying a sufficient condition under which full dishonesty, as previously defined, is no longer a Nash equilibrium.

Proposition 2. *With completely verifiable on-chain auditing enforcing off-chain audits, if*

$$S_{\text{au}} > \left(\frac{1 - p_{\text{au}}}{p_{\text{au}}}\right) r_{\text{au}}, \tag{3}$$

then (mutual) full dishonesty is not a Nash equilibrium.

[14] Note that our positive results will continue to hold for any $f_{\text{st}} \geq 1$, as higher frequency of audits reinforces the audit mechanism's robustness.

[15] This is a normalization, as we could alternatively replace the slashing penalty S_{au} with $S_{\text{au}} + r_{\text{au}}$.

Condition (3) guarantees the existence of a profitable deviation away from the dishonest equilibrium. Specifically, it implies that the slashed amount S_{au} is sufficiently large to make a false report of 1—as prescribed in the dishonest strategy—unprofitable. However, our goal is stronger than that: we wish to sustain an honest strategy as an equilibrium.

Definition 1. *We let $\sigma_i^\bullet$ denote the* honest strategy *of SP i during an audit epoch, i.e., $\sigma_i^\bullet = (s_i, e_i, \rho_i)$, where $s_i = 1$, $e_i^j = 1$ for all $j \in I \setminus \{i\}$, and $\rho_i^j = 1$ if and only if SP j has submitted a valid proof of storage to i.*

The honest strategy consists of two components that reflect the dual roles of each SP as a storage provider and an auditor. In the role of storage provider, the SP stores all assigned data chunks. As an auditor, the SP performs all assigned audits and reports their outcomes truthfully.

We assume that the network operates under low-level noise, i.e., there is a small probability $\epsilon > 0$ that any SP may submit an inaccurate proof due to minor errors or imperfections in the network.[16]

We now proceed to derive conditions under which the honest strategy profile constitutes the unique Nash equilibrium.

Theorem 1. *With completely verifiable on-chain auditing enforcing off-chain audits, if*

(i) slashing discourages reporting without auditing, i.e., $S_{\text{au}} \geq \left(\frac{1-p_{\text{au}}}{p_{\text{au}}}\right) r_{\text{au}} + \left(\frac{1}{\epsilon \cdot p_{\text{au}}}\right) c_{\text{au}}$;

(ii) audit rewards outweigh the cost of auditing, i.e., $r_{\text{au}} \geq \left(\frac{1}{1-\epsilon}\right) c_{\text{au}}$; and

(iii) storage rewards exceed costs, i.e., $r_{\text{st}} \geq c_{\text{st}}$;

then, mutual honesty, i.e., $\sigma^\bullet = (\sigma_1^\bullet, \ldots, \sigma_N^\bullet)$, is the unique Nash equilibrium.

Together, Conditions (i)–(iii) ensure that the expected payoff from honest behavior—both in auditing and data storage—strictly exceeds any potential gains from deviation. Furthermore, under these conditions, the honest strategy constitutes the unique equilibrium. Condition (i) guarantees that the slashing penalty is large enough to deter SPs from not carrying out audits while falsely reporting 1, regardless of whether a proof of storage was received.[17,18] Condition (ii) ensures that audit rewards are large enough to discourage providers

[16] This assumption reflects the practical reality that occasional inaccuracies can arise from benign sources such as hardware faults, timing issues, or other forms of operational noise.

[17] Condition (i) subsumes the inequality (3) in Proposition 2, which in this case guarantees that SPs will not report 1 in the absence of a valid proof.

[18] Note that Condition (i) becomes arbitrarily demanding as $\epsilon \to 0$, but that is because of a "verifier's dilemma" effect: if each SP expects all other SPs to properly store their data in equilibrium, and there is no chance of any SP making an error, then there is no incentive to actually conduct audits. In that case, we expect a partially-honest mixed strategy, in which SPs reduce their storage performance to optimally trade off relative to equilibrium auditing levels.

from merely reporting 0 without actually performing the audits. Finally, Condition (iii) is a baseline participation constraint, ensuring that acting as a storage provider is, at minimum, individually profitable.

3.3 Resilience to Coalitional Deviations

In this section, we analyze the robustness of the honest equilibrium in the presence of potential collusion among SPs aiming to maximize their collective payoff. We consider two distinct cases. In the first and stronger case—the *commitment* case—coalition members can commit in advance to a joint strategy from which no deviation is feasible ex post. This effectively allows the coalition to operate as a single coordinated entity, for example through a smart contract that enforces the agreed-upon strategy.[19] In the second, weaker case—the *no commitment* case—coalition members can deviate from the agreed-upon strategy. Here, the coalition merely exchanges information and agrees on a joint strategy ex ante, without any mechanism to bind members to the chosen strategy.

Definition 2. *An action profile σ is* δ-coalition–resistant with respect to coalition $J \subseteq \{1, \ldots, N\}$ *if for every joint deviation σ'_J of J, we have $\sum_{i \in J} u_i(\sigma) \geq \sum_{i \in J} u_i(\sigma'_J, \sigma_{-J}) - \delta$.*

In the commitment case, the maximum gain that a coalition of fewer than half the SPs can obtain by deviating from the honest strategy is bounded by the total audit cost incurred from internal auditing within the coalition.

Theorem 2. *Let J be a coalition of storage providers with $|J| < N/2$, and suppose that members of J have commitment power. Then, under Conditions (i)–(iii) of Theorem 1, the (mutually) honest strategy is $\left(|J|^2 \cdot c_{\text{au}}\right)$-coalition–resistant with respect to J.*

The main intuition behind Theorem 2 is that, as long as the coalition size remains below half of the total number of SPs—and under the appropriate conditions outlined in Theorem 1—it is always in the coalition members' best interest to fully store their assigned data. This inherently limits the potential gains from forming a coalition. Indeed, because the coalition members will be audited by a voting majority of SPs outside the coalition, the best joint deviation the coalition members can achieve is storing their data while not actively auditing each other.[20] But when the coalitional equilibrium self-enforces the storing of data,

[19] Such a smart contract could require each SP to stake a certain amount, which would be slashed if the SP deviates from the agreed-upon strategy.

[20] We might worry that SPs could try to free-ride by copying others' audit reports, for example if those reports are visible while being submitted through a public mempool. While in principle it is possible for this to induce a form of herding that would reduce the effective share of "independent" audit reports outside of a coalition, in practice it would require SPs to trust others to submit reports—and to make those reports accurately; SPs would be leaving themselves open to slashing otherwise. (Moreover, such trust would seem especially difficult to sustain when network participants are concerned about collusion.)

within-coalition auditing is not actually necessary—so there is no loss in terms of equilibrium data storage.[21] Importantly, this result does not depend on how the coalition's gains are allocated among its members.[22]

Moreover, if we additionally assume that coalition members do not have commitment power, then for any coalition J with $|J| < N/2$, no joint deviation can strictly improve the payoffs of all its members. In this case, the honest strategy remains robust even under coordinated deviations.

Definition 3. *An action profile σ is an* ℓ-strong equilibrium *if, for every subset J of at most $\ell \geq 0$ storage providers and for every joint deviation σ'_J of J, there exists at least one storage provider $i \in J$ such that $u_i(\sigma'_J, \sigma_{-J}) \leq u_i(\sigma)$.*

Our next result shows that for any coalition J with $|J| < N/2$, joint deviations cannot be sustained without commitment power.

Theorem 3. *Suppose that SPs do not have commitment power within coalitions. Then, for $\ell < N/2$, under Conditions (i)–(iii) of Theorem 1, the (mutually) honest strategy is the unique ℓ-strong equilibrium.*

Theorem 3 shows that when coalition members do not have commitment power, the honest equilibrium is robust to coalitions of less than half of the SPs. The intuition that underlies this result is that without the ability to commit to providing services when required, SPs within a coalition lack a strict incentive to submit valid proofs of storage ex post.

4 Strengthening the Protocol

Theorems 2 and 3 provide formal coalition-resistance guarantees. However, they are not the strongest guarantees possible. In particular, when coalition members have commitment power, a coalition J can still obtain an aggregate benefit of $|J|^2 \cdot c_{\text{au}}$. We show that while the current Shelby auditing protocol has this caveat, the protocol can be improved to provide a stronger coalition-resistance

[21] This implies that if a coalition of SPs develops a more efficient method to enforce data storage than the protocol's auditing mechanism, they can, in fact, locally outperform the protocol in maintaining honest storage.

[22] Note also that the preference for storing assigned data chunks over attempting on-the-fly reconstruction (Observation 1) continues to hold even in the presence of coalitions. Indeed, a coalition performing reconstruction still requires access to $\frac{2}{3}$ of all relevant chunks; thus, they benefit only from the subset of those chunks that are already stored by coalition members. Let x denote the fraction of assigned chunks stored by each member of a coalition J, and $\frac{|J|}{N}$ denote the relative size of the coalition. Then the savings in read cost per chunk from intra-coalition reads (which are assumed to be free) is $(1-x) \cdot \left(\frac{2}{3}\right) - (1-x) \cdot \left(\frac{2}{3} - x \cdot \left(\frac{|J|}{N}\right)\right)$; this is maximized when $x = \frac{1}{2}$, where it evaluates to $\frac{1}{4}\left(\frac{|J|}{N}\right)$, corresponding to a savings of at most 25%. Within the context of (1), a 25% reduction is negligible compared to the estimated 1600% overhead of on-the-fly reconstruction computed in Section 5.

guarantee, making the honest strategy an ℓ-strong equilibrium even given full commitment among coalition members. (This is the strongest notion of a coalition, under which all members always act to the joint benefit of the coalition: in the case of a distributed system, the commitment case can be thought of as if all coalition members are controlled by a single entity.) Beneficially, the proposed modification is minimal and preserves the structure of the original protocol.

Recall that an SP, in its role as an auditor, submits an audit report on-chain. This report consists of binary values only—1 for audits that received a successful response, and 0 otherwise. In addition, the auditor retains the successful audit responses for potential downstream audit-the-auditors inspection. We propose that the on-chain submission include an additional vector commitment over the set of successful audit responses.[23] This commitment—a fixed number of bytes when using Merkle trees—is added to the audit report; the underlying data itself is still retained locally by the auditor. The random inspection of audit reports will then be answered by both the audit response itself (the 1 KiB inclusion proof of the audited chunk) *and* an inclusion proof of the audit response verified against the additional embedded commitment in the report.

Proposition 3. *Assume Conditions (i)–(iii) of Theorem 1 hold and suppose the auditing protocol employs a vector commitment scheme for the reports. Then, for $\ell < N/2$, the (mutually) honest strategy is the unique ℓ-strong Nash equilibrium, even when coalition members have commitment power.*

The inclusion of a vector commitment scheme ensures that even within a coalition, a storage provider will not report a value of 1 without having actually received a valid proof of storage. This is because any attempt to generate the proof on-demand during the on-chain verification phase would invalidate the initial report, resulting in the provider being slashed. Thus, this modest modification to the audit protocol significantly strengthens its robustness to collusion.

5 Discussion

In this paper, we introduced a game-theoretic framework to analyze and prove incentive compatibility of the Shelby decentralized storage protocol. Our negative incentive result for the model with just off-chain audits (Proposition 1) highlights the importance of reasoning about incentives in these protocols: Even with a form of auditing in the system, incentive compatibility is not guaranteed without some way to enforce incentive alignment in the auditing process. Our positive results show that Shelby's occasional on-chain audit verification suffices to align auditor incentives, which in turn renders the full storage network incentive compatible (Theorems 1–3 and Proposition 3).

Our results highlight a simple blockchain mechanism-design principle: *infrequent but perfectly-verifiable checks can discipline cheaper, high-frequency off-chain interactions*. In Shelby, occasional on-chain verification makes truthful

[23] Vector commitments are already used in the auditing protocol, so our proposal does not introduce new cryptographic mechanism overhead.

auditing a best response; once audits are truthful, storage becomes incentive compatible as well. The point is architectural: when a protocol pairs (i) low-cost peer reporting with (ii) a scarce but certain report verification channel, incentive-aligned behavior can emerge at reasonable cost.

Back-of-the-Envelope Calibration. Using engineering-informed orders of magnitude we gathered from conversations with the SHELBY Engineering Team [30] ($r_{\mathrm{st}} \approx \$30/\mathrm{TB}/\mathrm{month}$, $c_{\mathrm{st}} \approx \$5/\mathrm{TB}/\mathrm{month}$, $c_{\mathrm{r}} \approx \$2/\mathrm{TB}$, $c_{\mathrm{au}} \approx \$10^{-7}$ per audit, $r_{\mathrm{au}} \approx \$5 \times 10^{-7}$ per audit, $p_{\mathrm{au}} \approx 2 \times 10^{-4}$ audit-the-auditor sampling, $S_{\mathrm{au}} \gtrsim \$1000$, $f_{\mathrm{st}} \approx 4$ audits/chunk/month), the qualitative takeaways are:

- **Decentralized storage is clearly worth doing.** Indeed, $r_{\mathrm{st}}/c_{\mathrm{st}} \approx 6$, so the basic participation constraint is comfortably satisfied.
- **Honest auditing beats "lazy negative" reporting.** As $r_{\mathrm{au}} \approx 5c_{\mathrm{au}}$, truthful auditing pays more than lazily reporting all-0, even allowing for small proof/measurement noise.
- **False "1" is a dominated strategy.** A fake 1 pays $r_{\mathrm{au}} \approx \$5 \times 10^{-7}$ but faces expected slashing $p_{\mathrm{au}} \cdot S_{\mathrm{au}} \approx \$0.2$; cheating is worse by a factor of 4×10^5. This margin remains substantial even if noise is as high as 1%.
- **On-the-fly reconstruction remains uneconomic.** With $k = 10$ and $f_{\mathrm{st}} \approx 4$, an auditee would need at least 40 chunk reads per month to mimic honest storage; $40c_{\mathrm{r}}/c_{\mathrm{st}} \approx 16$ makes this highly unattractive in expectation.

These back-of-the-envelope checks are not "tight bounds," but they show that the SHELBY design sits in a regime where the honest strategy is comfortable.

Incentivizing Whistle-Blowing is Complementary. Many protocols (including SHELBY) consider bounties for submitting forensic proofs of misbehavior. Our guarantees do not rely on bounties; we show that at least in the case of SHELBY, a whistleblower reward is not necessary for incentive alignment. That said, in practice, modest bounties can further deter collusion and speed fault detection, with standard griefing/DoS safeguards on submission.

Participation Elasticities and Slashing. In Theorem 1, the key condition—assuming rewards are high enough to cover costs—is that the slashing penalty for incorrect audit reports, S_{au}, is sufficiently large. We might then wonder: why not set the penalty S_{au} at some extremely high value?[24]

While in theory arbitrarily high S_{au} would allow us to reduce the required on-chain audit probability p_{au} to near zero, there are several practical concerns with such an approach. First, in order for the system to effect slashing, SPs must stake value equal to S_{au}, so if S_{au} is extremely high, then the capital requirements for participation may be prohibitive for smaller SPs, limiting decentralization. Moreover, even honest SPs who are able to cover the staking capital requirements may be disinclined to commit such large amounts if the system rewards are not

[24] We thank a reviewer for raising this question, which, on reflection, we think should be addressed more commonly in protocol design.

large enough to compensate for putting their capital at (even small) risk. Thus, in general, it is important to trade off between the slashing level S_{au} and the probability of auditing p_{au}; and more broadly, we expect that in practice the inequality Condition (i) in Theorem 1 should ideally be relatively tight, in the sense that slashing penalties should be set at a level to discourage but not overkill reporting without auditing.[25]

Beyond Storage: DePIN and Data Availability. While the incentive design and analysis presented in this paper are rooted in the context of decentralized data storage, the same template applies in other contexts where work is off-chain but attestable, as in, for example, sensor readings, bandwidth/compute serving, or rollup data-availability sampling. In each case, peer reports carry most of the load, and a small budget of certain checks (cryptographic openings, ZK spot-checks, or L1 verifications) enforces honesty. The choice is not "all on-chain vs. all off-chain," but a calibrated split with a credible hammer.

Limitations and Future Work. One limitation of our analysis is the focus on Nash equilibrium as the solution concept. It remains an open question whether there are stronger implementation guarantees, such as dominant-strategy incentive compatibility, for auditing in decentralized storage networks.

Additionally, our model is static. To some degree, we view this as a feature, as it means we obtain within-period incentive guarantees—without the need to appeal to reputation or other dynamic incentives, or to consider the balance between rewards and time discounting. However, a dynamic setting may create a richer strategy space that allows for behavior not captured in our model.

Our analysis also assumes each SP corresponds to a single player in the game. In deployment, incentive compatibility must compose with Sybil defenses (here, staking requirements). Quantifying the combined margin is an open task.

And meanwhile, the inspection rate p_{au} trades off against L1 cost and settlement delay. Our calibration works for Aptos; systems evaluation mapping feasible p_{au} to specific chains (throughput, gas, time-to-finality) would turn our qualitative guidance into deployment playbooks on those chains.

Coda. Overall, we view our results as *sufficient* conditions and a design blueprint: pair cheap peer audits with a small budget of certain checks, calibrate rewards and slashing so the expected value of cheating is deeply negative, and (optionally) bind audit artifacts with vector commitments to harden against collusion—then honest storage and truthful auditing emerge as the unique equilibrium.

[25] As a side note, a second concern, reminiscent of the Becker [6] theory of crime, is that if slashing penalties are excessive relative to what is needed to provide proper incentives, then SPs who conclude they are at risk of slashing may decide to compound their malbehavior, under the theory that they have nothing left to lose.

Acknowledgments

We thank Pranav Garimidi, Guru Vamsi Policharla, Tim Roughgarden, and seminar participants at a16z crypto for helpful comments, as well as the FC referees, and especially the FC shepherd.

Part of this work was conducted while Kominers was visiting the Technological Innovation, Entrepreneurship, and Strategic Management (TIES) Group at the MIT Sloan School of Management; he greatly appreciates their hospitality.

Disclosures

Goren is employed by Aptos Labs and is a part of the team working on the Shelby protocol.

Kominers is a Research Partner at a16z crypto, and part of this work was conducted while Crystal was a Research Intern at a16z crypto; a16z crypto reviewed a draft of this article for compliance prior to publication and is an investor in various crypto projects, including Aptos (for general a16z disclosures, see https://www.a16z.com/disclosures/). Notwithstanding, the ideas and opinions expressed herein are those of the authors, rather than of a16z or its affiliates.

Kominers also holds digital assets, including both fungible and non-fungible tokens, and advises a number of companies on marketplace and incentive design, and serves as an expert on related matters.

References

1. Al-Bassam, M.: LazyLedger: A distributed data availability ledger with client-side smart contracts (2019), https://arxiv.org/abs/1905.09274 1, 3
2. Al-Bassam, M., Sonnino, A., Buterin, V.: Fraud and data availability proofs: Maximising light client security and scaling blockchains with dishonest majorities (2018), https://arxiv.org/abs/1809.09044 3
3. Alchian, A.A., Demsetz, H.: Production, information costs, and economic organization. American Economic Review **62**(5), 777–795 (1972) 4
4. Amazon Web Services: Amazon S3 pricing (2025), https://aws.amazon.com/s3/pricing/, accessed: 2025-08-05 8
5. Aptos Labs: The Aptos blockchain: Safe, scalable, and upgradeable web3 infrastructure (2022), https://aptosfoundation.org/whitepaper/aptos-whitepaper_en.pdf, whitepaper v1.0 5
6. Becker, G.S.: Crime and punishment: An economic approach. Journal of Political Economy **76**(2), 169–217 (1968) 16
7. Benet, J.: IPFS – content addressed, versioned, P2P file system (2014), https://arxiv.org/abs/1407.3561 1
8. Breidenbach, L., Chacin, C., Chan, B., Coventry, A., Ellis, S., Juels, A., Koushanfar, F., Miller, A., Magauran, B., Moroz, D., Nazarov, S., Topliceanu, A., Tramèr, F., Zhang, F.: Chainlink 2.0: Next steps in the evolution of decentralized oracle networks (2021), https://research.chain.link/whitepaper-v2.pdf, whitepaper v2 4

9. Catalano, D., Fiore, D.: Vector commitments and their applications. In: Kurosawa, K., Hanaoka, G. (eds.) Public-Key Cryptography – PKC 2013. Lecture Notes in Computer Science, vol. 7778, pp. 55–72. Springer (2013). https://doi.org/10.1007/978-3-642-36362-7_5 5
10. Crust Network: Crust: Technical white paper v1.9.9 (2020), https://gw.crustapps.net/ipfs/QmP9WqDYhreSuv5KJWzWVKZXJ4hc7y9fUdwC4u23SmqL6t 1
11. Crust Network: Crust economy whitepaper (2021), https://crust.network/download/ecowhitepaper_en.pdf 4
12. Dai, Y., Blanchard, M., Jaillet, P.: Non-monetary mechanism design without distributional information: Using scarce audits wisely (2025), https://arxiv.org/abs/2502.08412 4
13. Danezis, G., Giuliari, G., Kokoris-Kogias, E., Legner, M., Smith, J.P., Sonnino, A., Wüst, K.: Walrus: An efficient decentralized storage network (2025), https://arxiv.org/abs/2505.05370, whitepaper v2 1, 3
14. Doan, T.V., Psaras, Y., Ott, J., Bajpai, V.: Toward decentralized cloud storage with IPFS: Opportunities, challenges, and future considerations. IEEE Internet Computing **26**(6), 7–15 (2022) 3
15. Du, Y., Duan, H., Zhou, A., Wang, C., Au, M.H., Wang, Q.: Enabling secure and efficient decentralized storage auditing with blockchain. IEEE Transactions on Dependable and Secure Computing **19**(5), 3038–3054 (2021) 4
16. Eigen Labs: EigenDA spec (2025), https://layr-labs.github.io/eigenda/, protocol specification (mdBook), last updated 2025-02-22 1, 3
17. Estornell, A., Chen, Y., Das, S., Liu, Y., Vorobeychik, Y.: Incentivizing recourse through auditing in strategic classification. In: Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence. pp. 400–408 (2023) 4
18. Google Cloud: Google Cloud storage pricing (2025), https://cloud.google.com/storage/pricing#network-pricing, accessed: 2025-08-05 8
19. Goren, G., Hariri, A., Hartley, T.D.R., Kappiyoor, R., Spiegelman, A., Zmick, D.: Shelby: Decentralized storage designed to serve (2025), https://arxiv.org/abs/2506.19233, whitepaper v1.0 2, 4, 8
20. Ihle, C., Trautwein, D., Schubotz, M., Meuschke, N., Gipp, B.: Incentive mechanisms in peer-to-peer networks—a systematic literature review. ACM Computing Surveys **55**(14s), Art. 308 (2023) 4
21. Li, C., Xu, M., Zhang, J., Guo, H., Cheng, X.: SoK: Decentralized storage network. High-Confidence Computing **4**(3), 100239 (2024) 2
22. Lundy, T., Wei, A., Fu, H., Kominers, S.D., Leyton-Brown, K.: Allocation for social good: Auditing mechanisms for utility maximization. In: Proceedings of the 2019 ACM Conference on Economics and Computation. pp. 785–803. ACM (2019) 4
23. Merkle, R.C.: Protocols for public key cryptosystems. In: Proceedings of the 1980 IEEE Symposium on Security and Privacy. pp. 122–134. IEEE (1980). https://doi.org/10.1109/SP.1980.10006 5
24. Microsoft Azure: Azure blob storage pricing (2025), https://azure.microsoft.com/en-us/pricing/details/storage/blobs/, accessed: 2025-08-05 8
25. Milionis, J., Ernstberger, J., Bonneau, J., Kominers, S.D., Roughgarden, T.: Incentive-compatible recovery from manipulated signals, with applications to decentralized physical infrastructure. In: Proceedings of the International Conference on Financial Cryptography and Data Security (Forthcoming, 2026), https://arxiv.org/abs/2503.07558 4
26. Protocol Labs: Filecoin: A decentralized storage network (2017), https://filecoin.io/filecoin.pdf, whitepaper v1.0 1, 3

27. Rahman, D.: But who will monitor the monitor? American Economic Review **102**(6), 2767–2797 (2012) 4
28. Reed, I.S., Solomon, G.: Polynomial codes over certain finite fields. Journal of the Society for Industrial and Applied Mathematics **8**(2), 300–304 (1960) 7
29. Roth, R.M.: Introduction to Coding Theory. Cambridge University Press (2006) 7
30. Shelby Engineering Team: Parameter estimates for the Shelby protocol. Personal communication (2025), conversations with engineers at Aptos Labs and Jump Crypto, Aug–Sep 2025 8, 15
31. Shelby Foundation: Shelby: Decentralized storage protocol (2025), https://shelby.xyz/, accessed: 2025-08-05 2, 8
32. Shen, Z.S., Cheng, K., Lee, P.P.C.: A survey of the past, present, and future of erasure coding for storage systems. ACM Transactions on Storage **21**(1), Art. 4 (2025). https://doi.org/10.1145/3708994 7
33. Storj Labs, Inc.: Storj V3: Technical white paper (2018), https://static.storj.io/storjv3.pdf, whitepaper v3.1 (updated 2024-05-16) 1, 3
34. Vakilinia, I., Wang, W., Xin, J.: An incentive-compatible mechanism for decentralized storage network. IEEE Transactions on Network Science and Engineering **10**(4), 2294–2306 (2023) 4
35. Vorick, D., Champine, L.: Sia: Simple decentralized storage (2014), https://sia.tech/whitepaper.pdf, whitepaper 1, 3
36. Wang, X., Azouvi, S., Vukolić, M.: Security analysis of Filecoin's expected consensus in the Byzantine vs. honest model (2023), https://arxiv.org/abs/2308.06955 3
37. Williams, S., Diordiiev, V., Berman, L., Raybould, I., Uemlianin, I.: Arweave: A protocol for economically sustainable information permanence (2019), https://www.arweave.org/yellow-paper.pdf, yellowpaper draft 1 1, 3
38. Wu, Z., Lin, B., Nikiforakis, N., Balasubramanian, A.: Degrees of decentralized freedom: Comparing modern decentralized storage platforms. In: Proceedings of the 2025 9th Network Traffic Measurement and Analysis Conference (TMA). p. Art. 5. IEEE (2025) 3

A Proofs Omitted from the Main Text

A.1 Proof of Proposition 1

We first show that mutual full dishonesty constitutes a Nash equilibrium. When all SPs play σ°, each SP i receives

$$u_i(\sigma_i^\circ, \sigma_{-i}^\circ) = r_{\text{st}} + (N-1)r_{\text{au}}.$$

This strategy incurs no storage or audit cost, while maximizing rewards. Therefore, no SP has an incentive to deviate.

To show uniqueness, suppose there were an alternative Nash equilibrium in which some SP i assigns an audit score of $\mathsf{0}$ to SP j, i.e., $\rho_i^j = \mathsf{0}$. In that case, SP i could profitably deviate by setting $\rho_i^j = \mathsf{1}$, thereby gaining an additional r_{au} without incurring any further cost (indeed, reporting $\rho_i^j = \mathsf{1}$ is strictly dominant). Therefore, in any equilibrium, all audit vectors must consist of $\mathsf{1}$s: $\rho_i^j = \mathsf{0}$ for all i and j.

Given that SPs receive full audit rewards regardless of actual auditing, and that auditing incurs a cost, it must be the case that $e_i^j = 0$ for all i and j in equilibrium. Similarly, storing data incurs cost c_{st} without affecting rewards if all audit reports are positive, implying $s_i = 0$ for all i. Thus, mutual full dishonesty is the unique pure strategy Nash equilibrium in the internal audit game. □

A.2 Proof of Proposition 2

Under mutual full dishonesty, each SP i follows the strategy

$$\sigma_i^\circ(s_i, e_i, \rho_i) = (0, 0, 0, \ldots, 0, 1, 1, \ldots, 1).$$

The payoff for SP i associated with $(e_i^j, \rho_i^j) = (0, \mathsf{1})$ is given by

$$(1 - p_{\text{au}})r_{\text{au}} - p_{\text{au}}S_{\text{au}},$$

while the payoff from setting $(e_i^j, \rho_i^j) = (0, \mathsf{0})$ is 0. Thus, deviating to $(0, 0)$ will be profitable if

$$S_{\text{au}} > \left(\frac{1 - p_{\text{au}}}{p_{\text{au}}}\right) r_{\text{au}};$$

in this case, mutual dishonesty would no longer be a Nash equilibrium. □

A.3 Proof of Theorem 1

We begin by showing that employing the honest strategy is an equilibrium. Assume that each storage provider (SP) j stores its assigned data (i.e., $s_j = 1$); we first show that no other SP i finds it profitable to deviate from the honest strategy when choosing their audit and reporting actions (e_i^j, ρ_i^j).

Let $u_i^j(e_i^j, \rho_i^j \mid s_j = 1)$ denote SP i's expected audit utility from choosing whether to audit SP j and what to report, given that SP j stores its assigned data. Then

$$\begin{aligned}
u_i^j(\sigma_i^\bullet \mid s_j = 1) &= \epsilon \cdot 0 + (1-\epsilon) r_{\text{au}} - c_{\text{au}} \\
u_i^j(0, \mathsf{1} \mid s_j = 1) &= \epsilon \left[(1-p_{\text{au}}) r_{\text{au}} - p_{\text{au}} S_{\text{au}}\right] + (1-\epsilon) r_{\text{au}} \\
u_i^j(0, \mathsf{0} \mid s_j = 1) &= 0 \\
u_i^j(1, \mathsf{0} \mid s_j = 1) &= -c_{\text{au}}.
\end{aligned}$$

To deter the deviation $(0, \mathsf{1})$, we require

$$u_i^j(\sigma_i^\bullet \mid s_j = 1) \geq u_i^j(0, \mathsf{1} \mid s_j = 1);$$

this implies

$$S_{\text{au}} \geq \left(\frac{1-p_{\text{au}}}{p_{\text{au}}}\right) r_{\text{au}} + \left(\frac{1}{\epsilon \cdot p_{\text{au}}}\right) c_{\text{au}},$$

which is Condition (i).

To deter the deviation $(0, \mathsf{0})$, i.e., not auditing and always reporting $\mathsf{0}$, we require

$$u_i^j(\sigma_i^\bullet \mid s_j = 1) \geq u_i^j(0, \mathsf{0} \mid s_j = 1),$$

which leads to

$$r_{\text{au}} \geq \left(\frac{1}{1-\epsilon}\right) c_{\text{au}},$$

which is Condition (ii).

Next consider deviations in which SP i chooses not to store its assigned data, i.e., $s_i = 0$. To render this a non-profitable deviation, it must be that any auditing SP j truthfully reports $\mathsf{0}$ if they detect an invalid or a missing proof. This will be the case if

$$(1-p_{\text{au}}) r_{\text{au}} - p_{\text{au}} S_{\text{au}} - c_{\text{au}} \leq -c_{\text{au}} \implies S_{\text{au}} \geq \left(\frac{1-p_{\text{au}}}{p_{\text{au}}}\right) r_{\text{au}},$$

which is implied by Condition (i).

Given this setup, if i deviates, then i will be audited truthfully by other SPs and will not receive storage rewards, as a majority of SPs will report $\mathsf{0}$ against it. Consequently, as long as

$$r_{\text{st}} - c_s \geq 0,$$

SP i has an incentive to store its assigned data; when this condition—which is equivalent to Condition (iii)—is satisfied along with Conditions (i) and (ii), honest behavior constitutes a Nash equilibrium.

We now establish uniqueness. For the sake of seeking a contradiction, assume that there exists an equilibrium in which some SP i does not store its assigned data. By Conditions (i) and (ii), no other SP will report a value of $\mathsf{1}$ for i, implying that i will not receive their storage reward r_{st}. Then, by Condition

(iii), i has a strict incentive to deviate and store its assigned data. Thus, it follows that in any equilibrium, all SPs must store their assigned data. Applying Conditions (i) and (ii) again, we then see that each SP will conduct their assigned audits, and report truthfully. Therefore, the honest strategy profile is the unique equilibrium. □

A.4 Proof of Theorem 2

With $|J| < N/2$, Conditions (i)-(iii) ensure that SPs outside the coalition follow the honest strategy. In particular, by Conditions (i) and (ii), and the reasoning in the proof of Theorem 1, any SP $i \notin J$ will report a value of 1 for another SP only if they have performed the audit and obtained a valid proof. Therefore, any SP within the coalition that does not store a data chunk will be unable to secure a majority of 1 votes for that chunk, regardless of the coalition's strategy. As a result, the storage-related payoff for any SP that does not store its assigned data will be at most 0, whereas storing its assigned data yields a payoff of $r_{\text{st}} - c_{\text{st}}$, which is positive by Condition (iii). Hence, in any equilibrium, all SPs must store their assigned data.

While storing data, coalition members may coordinate to report 1 scores for each other without actually performing audits. When these 1 scores are themselves audited via the on-chain audit mechanism, the assigned SP will be required to submit a valid proof of storage. Given that coalition members have full commitment power, they can pre-commit to providing such proofs when requested. Thus, assuming that no SP within the coalition conduct audits on other coalition members, the maximum additional payoff gained will be $|J|^2 \cdot c_{\text{au}}$. □

A.5 Proof of Theorem 3

Following the reasoning in the proof of Theorem 2, for any $|J| < N/2$, Conditions (i)-(iii) ensure that service providers (SPs) outside the coalition adhere to the honest strategy. As a result, any SP that fails to store its assigned data receives a storage-related payoff of 0, whereas storing the data yields a payoff of $r_{\text{st}} - c_{\text{st}}$, which is positive by Condition (iii). Therefore, in any equilibrium, all SPs must store their assigned data.

However, when coalition members lack commitment power, deviating from the honest strategy—by falsely reporting 1 scores for one another without performing actual audits—ceases to be profitable because there is no guarantee that valid proofs will be generated and submitted when required. Consequently, for any $|J| < N/2$, the honest strategy constitutes the unique $|J|$-strong equilibrium in the no-commitment case. □

A.6 Proof of Proposition 3

In the case in which coalition members lack commitment power, the result follows directly from Theorem 3.

Now consider the case in which coalition members have commitment power. The vector commitment scheme guarantees that any report of 1 without a corresponding valid proof at the time of reporting will trigger slashing during on-chain verification. (This is because the vector commitment scheme effectively creates a cryptographic timestamp at the time of proof generation.) As a result, any attempt to construct a proof at a later stage will not constitute a valid proof and will therefore be penalized. Consequently, any coordinated attempt within the coalition to submit 1 scores without actually performing the audits carries the risk of being slashed. Therefore, under the vector commitment scheme and conditions (i)-(iii), coalition members have no profitable deviation from the honest strategy. □