Response to Reviewers

Manuscript Title: *dcChecker: Efficient Deterministic and Continuous Checking for Critical Data Integrity in Decentralized Storage*
Previous Submission: ESORICS 2025
Submission ID: 72

We sincerely thank the reviewers of our previous submission to ESORICS 2025 for their constructive and detailed feedback. Based on their comments, we have revised our manuscript extensively. Below we provide a point-by-point response to each review, including the specific modifications made to the manuscript and the sections where they appear.

Reviewer #1

Comment 1.1: *"The application scenario and the attack scenario are not clearly stated upfront."*

Response: We have added a dedicated **Motivation** section (Section 1.A) that explicitly describes the application context, including whether data is stored redundantly, encrypted, and how updates or provider departure is handled. Additionally, we introduce a detailed **Threat Model** in Section 1.B to clarify attacker capabilities and assumptions, specifying whether the storage providers are honest-but-curious or malicious, and what kinds of attacks are considered.

Comment 1.2: *"The authors criticize probabilistic protocols but don't clearly show the advantage of determinism."*

Response: We now explain the necessity of deterministic checking by comparing the failure probability of probabilistic schemes and their limitations in critical scenarios (Section 6.A). We also highlight how deterministic checking offers stronger guarantees for critical data integrity, and how our approach improves over state-of-the-art schemes such as CBDIC (Section 6.B).

Comment 1.3: *"Language needs revision; many sentences are vague."*

Response: We have thoroughly revised the manuscript to improve clarity, with particular attention to ambiguous or vague expressions in the introduction and related work sections.

Reviewer #2

Comment 2.1: *"Grammar and sentence structure need improvement."*

Response: We carefully revised the grammar throughout the paper. Sentences like "Current work just considers the second issue and ignore the first issue" have been rephrased for clarity, such as "Most existing work focuses only on continuous checking and overlooks the need for determinism."

Comment 2.2: *"The novelty and necessity of combining deterministic and continuous checks is not sufficiently justified."*

Response: We now elaborate in Section 1.A and 6.A why both properties are needed and how dcChecker uniquely fulfills both requirements. We also quantitatively compare our approach against prior work in Section 6.B.

Comment 2.3: *"Attacks like tag forgery, data deletion, replacement, and leakage should be defined and explained."*

Response: We expanded Section 6.4 significantly and added detailed definitions of these attacks in the **Threat Model** (Section 1.B), linking them clearly to the protection goals of our protocol.

Reviewer #3

Comment 3.1: *"Is deterministic checking necessary when probabilistic detection is already very strong (e.g., 2^-128)?"*

Response: In Section 6.A, we analyze detection probabilities in existing probabilistic schemes and show that in some cases, especially in settings without tight audit frequencies, the effective detection probability is much lower than claimed. We provide examples where even a single missed attack could be catastrophic, justifying the need for deterministic guarantees.

Comment 3.2: *"The use of blockchain is unnecessary; signatures may suffice."*

Response: We clarify in Section 3.A and 4.A that blockchain is not merely used for message relay but for **tamper-proof logging** of proof verification results, ensuring transparency and dispute resolution among **multiple parties** (data owners, verifiers, auditors). This is something digital signatures alone cannot achieve in a decentralized setting.

Comment 3.3: *"Performance is low (10MB verification takes 10 seconds), consider*

comparing to transmission time."

Response: We revised Section 6 to explain the experiment setup and measurement more clearly. We show that while initial verification latency is around 10 seconds for a 10MB file, our scheme scales better for larger data sizes due to its aggregation and batch verification capabilities. We also compare this to actual network transmission latency in typical decentralized systems to show the practicality of our approach.