Two-Forest Publisher Maintenance and Transcript-Free Recovery for Incremental Decentralized Data Archival

August 30, 2026

Abstract

We study incremental decentralized data archival for a database of at most N blocks in $\mathbb{F}_{2^B}$, initialized with n_0 blocks and extended by one-block Update calls. For a slack $\epsilon = \epsilon(\lambda) \in (0,1)$ with polynomially bounded $1/\epsilon$, we give an explicit Setup–Init–Join–Update–Audit construction based on tagged dyadic chunks and an additive-evaluation $[Rn, n]$ MDS code, where $R = \Theta(1/\epsilon)$ and $R2^{\lceil \log_2 N \rceil} \le 2^B$. Its initialization prefix is frozen in one forest and the post-initialization suffix is maintained in a second forest. For every $1 \le U \le N - n_0$, node-independent publisher maintenance over the first U Updates is $O(BU \log N/\epsilon)$ in the simplified block-arithmetic model, including every public-digest serialization and without a relation between n_0 and U; equivalently, the iDDA amortized cost is $O(B \log N/\epsilon)$ per Update. Publisher storage is $O(BN/\epsilon)$; the coding bit-work bound replaces B by $M(B) + B$, where $M(B)$ is the bit cost of field multiplication. Setup, Init, and identity-dependent serving are excluded from this maintenance measure.

For recovery, let S be the advertised node capacity in blocks, let μ be the number of pairwise-distinct audited identities, let κ be the number of challenges per sampled chunk, let Δ bound graph indegree, and let λ_{vc} be the commitment length. Assume the local verified-extraction contract for a succinct depth-robust-graph interface, persistent cache-hidden domain-separated random oracles, quasi-polynomial vector-commitment binding, polynomial-size public encodings, and polynomially bounded μ, N, κ, R. Take a public $w = \omega(1)$ satisfying $w \log_2 n_0 \ge 1$ and $2(\Delta + 1)\kappa w \log_2 N \le S$. Then the same construction has transcript-free paired recovery provided $\epsilon\kappa = \omega(\log \lambda)$, $S = \omega(\kappa \log^2 \lambda)$, $n_0 \ge \max\{S, \lambda\}$, $\lambda_{\mathrm{vc}} = \omega(\log \lambda)$, and $B = \omega(\lambda_{\mathrm{vc}} \log N)$. Its compressor may use the experiment transcript, whereas its quasi-polynomial extractor receives neither that transcript nor the original database. Their only shared randomness is a finite uniform string, their advice has at most $B \max\{n - (1-\epsilon)\mu S, 0\}$ bits, and

$$\Pr[\mathsf{Pass} \wedge (DB_{\mathrm{ext}} \neq DB^*)] \le \mathrm{negl}(\lambda).$$

The theorem concerns recoverability with paired advice and does not assert a replication bound.

Note. This paper was fully AI generated through a harness created by Richard Peng that discovers open problems and runs them through an automated research pipeline.

Contents

1 Introduction

An append-only public database can eventually outgrow the storage available to an ordinary participant. Keeping a full replica at every participant then raises the barrier to entry, whereas merely distributing coded fragments does not explain which fragments remain available when participants are untrusted. Incremental decentralized data archival (iDDA), formulated by Shi, Silver, and Mu [SSM26], brings these issues into one cryptographic interface. A publisher initializes a database, storage nodes join with a fixed space budget, the publisher appends blocks, and periodic audits test the nodes' evolving shards. The central recovery requirement is aggregate: identities that pass their audits should collectively account for useful database information in proportion to their claimed space, even when no single identity can store the database.

Several established lines of work capture parts of this problem. Rabin's information-dispersal scheme gives threshold reconstruction from coded fragments [Rab89]. Proofs of retrievability couple successful audits to extraction of an outsourced file, while provable data possession supplies efficient sampled possession checks [JJ07, ABC+07]. Dynamic proofs of retrievability support updates and extraction, but provision one server for the whole file [CKW13, SSP13]. Thus these are nearby foundations, or full-server restrictions of the archival problem, rather than results about aggregating many permissionless low-space identities.

Other systems are natural restricted versions of decentralized archival. Permacoin ties mining rewards to preserving coded portions of a large static public archive [MJS+14]. Pérard et al., Kadhe et al., and Raman and Varshney reduce the history stored by an individual blockchain node and reconstruct from sufficiently many available or honest participants [PLBD18, KCR19, RV21]. Permacoin is static, while the coded-history guarantees are stated in their respective availability or corruption models. None of these works gives recovery proportional to the space claimed by an arbitrary set of audit-passing pseudonyms.

Our direct same-problem comparison is the iDDA construction of Shi, Silver, and Mu [SSM26]. It provides the five-protocol Setup–Init–Join–Update–Audit interface and treats recoverability and replication security as separate objectives. In its random-oracle construction, publisher computation is $B\exp(O(1)/\epsilon)\log N$ per Update, amortized over the prescribed $N-n_0$ Updates. This paper addresses the exponential dependence on the slack in that publisher-maintenance bound.

Our contribution. For a database initialized with n_0 field elements of B bits and growing to at most N elements, we construct the five-protocol scheme TwoForest. The result permits a varying slack $\epsilon = \epsilon(\lambda)$ with polynomially bounded reciprocal. For every realized prefix of U post-initialization Updates, the scheme maintains the publisher state and complete public digest using

$$O\left(\frac{BU\log N}{\epsilon}\right)$$

node-independent work. Equivalently, in the amortized terminology of the direct comparison, publisher maintenance is $O(B\log N/\epsilon)$ per Update: its dependence on $1/\epsilon$ is linear rather than exponential. The prefix guarantee imposes no relation between n_0 and U and is not a worst-case per-call bound. It excludes one-time Setup and Init and identity-dependent serving, and it includes every complete public digest. Publisher storage is $O(BN/\epsilon)$, and the corresponding coding bit work over the prefix is $O((M(B)+B)U\log N/\epsilon)$, where $M(B)$ denotes the bit complexity of multiplication in $\mathbb{F}_{2^B}$.

Under the recovery hypotheses stated below, the paired compressor and the quasi-polynomial extractor TwoForestRecovery additionally guarantee that μ pairwise-distinct identities which

all pass their audits can be combined with at most

$$B \max\{n - (1-\epsilon)\mu S, 0\}$$

bits of advice to recover the current n-block database, except with negligible joint error: the probability that all audits pass but the extracted database is wrong is negligible. The extractor receives neither the audit transcript nor the original database. This is a paired recoverability statement: it does not establish replication security or a physical-space lower bound.

We now state the strongest, variable-slack form of the result. All asymptotic conditions are for $\lambda \to \infty$, and all logarithms are base two.

Theorem 1.1 (Two-forest publisher and recovery improvement)**.** *Let $\epsilon = \epsilon(\lambda) \in (0,1)$ satisfy $1/\epsilon \le \lambda^{O(1)}$, let $N > n_0$, and put*

$$\widehat{N} = 2^{\lceil \log_2 N \rceil}, \qquad k_{\mathrm{rec}} = \left\lceil \log_2 \frac{1}{\epsilon} \right\rceil, \qquad \bar{\epsilon} = 2^{-k_{\mathrm{rec}}},$$

$$\eta = \frac{\bar{\epsilon}}{40}, \qquad \vartheta = 1 - 13\eta, \qquad \zeta_0 = \frac{\bar{\epsilon}}{8}, \qquad R = 2^{\lceil \log_2(480\mathrm{e}/\bar{\epsilon}) \rceil}.$$

Assume $R\widehat{N} \le 2^B$, $\lambda_{\mathrm{vc}} \le B$, and the publisher-admissibility conditions of Definition 4.1. Then the five-protocol scheme $\Pi_{2\mathrm{F}} = \textsc{TwoForest}$ is honestly correct. For every $U \in [1, N - n_0]$, the first U post-Init Update calls use

$$O\left(\frac{BU \log N}{\epsilon}\right)$$

simplified node-independent publisher work. This includes each complete public digest, excludes one-time Setup and Init and identity-dependent serving, and imposes no horizon condition relating n_0 to U. Publisher storage is $O(BN/\epsilon)$, the coding bit work is $O((M(B)+B)U \log N/\epsilon)$, and a full-codeword Merkle tree has depth $O(\log N + \log(1/\epsilon))$. Thus, in the iDDA amortized accounting, the simplified cost is $O(B \log N/\epsilon)$ per Update; the claim is not a worst-case per-call bound.

Suppose in addition that the family is recovery-admissible in the sense of Definition 4.1, that the local verified-extraction contract of Definition 4.7 holds uniformly at every legal current length, and that

$$\epsilon\kappa = \omega(\log\lambda), \quad 1 \le \mu \le \lambda^{O(1)}, \quad S = \omega(\kappa \log^2 \lambda), \quad n_0 \ge \max\{S, \lambda\},$$
$$w = \omega(1), \qquad \lambda_{\mathrm{vc}} = \omega(\log\lambda), \qquad B = \omega(\lambda_{\mathrm{vc}} \log N),$$

and that the μ challenge identities are pairwise distinct. Then the two entry points of TwoForestRecovery *are total and have the signatures*

$$\mathcal{C}^{\mathcal{A}}(1^\lambda, S, \mu, tr; \rho) \longrightarrow DB_{\mathrm{short}},$$

$$\mathcal{E}^{\mathcal{A}_2(st_{\mathcal{A}})}(1^\lambda, n, S, \mu, \varphi_n, DB_{\mathrm{short}}; \rho) \longrightarrow DB_{\mathrm{ext}} \text{ or } \perp.$$

Here ρ is finite uniform shared randomness independent of the receiver experiment. The extractor receives neither tr nor DB^*, may face arbitrary adaptive dependence of later audits on earlier audits, and runs in time quasi-polynomial in $(\lambda, t_{\mathcal{A}})$. For advice emitted by the paired compressor in the same public, continuation, persistent-oracle, and shared-coin context,

$$|DB_{\mathrm{short}}| \le B \max\{n - (1-\epsilon)\mu S, 0\},$$

and

$$\Pr[\mathsf{Pass} \wedge (DB_{\mathrm{ext}} \ne DB^*)] \le \mathrm{negl}(\lambda).$$

At these parameters an honest node state has $O(BS)$ bits. If an Update refreshes d honest participants, its separately charged identity-dependent output is $O(dBS)$ bits and its complete explicit output, including the one public digest, is $O((d+1)BS)$ bits. Here the publisher hierarchy is updated in place and is not reserialized as output. No correctness assertion is made for unpaired advice, and no replication probability conclusion is part of the theorem.

Proof ideas and scope. The publisher construction freezes the initialized prefix in one tagged dyadic forest and maintains only the appended suffix in a second binary-counter forest. An additive-evaluation MDS code with redundancy $R = \Theta(1/\epsilon)$ makes equal-size encoded chunks recursively mergeable; the binary-counter accounting makes each appended block participate in only $O(\log N)$ merge levels, yielding $O(BR \log N)$ average work. For recovery, accepted audits yield candidate codeword symbols, an accepting-spine argument handles adaptive audit dependence without giving the transcript to the extractor, and collision expansion followed by error-and-erasure decoding completes each chunk. These ingredients prove the stated paired-advice bound. The construction of Shi, Silver, and Mu also proves a replication-security result [SSM26], whereas Theorem 1.1 assumes distinct challenged identities and proves recoverability only; consequently, the publisher improvement is not an overall strengthening of their theorem.

Organization. Section 2 fixes the computational model, the recovery experiment, and the accounting conventions. Section 3 explains the publisher and recovery arguments. Section 4 constructs the five-protocol scheme and proves Theorem 1.1.

2 Preliminaries

Notation. All logarithms are base two. For integers $a \le b$, write $[a, b] = \{a, a+1, \ldots, b\}$ and $[a, b) = \{a, a+1, \ldots, b-1\}$; in particular, $[m] = [0, m)$. The symbol $\|$ denotes concatenation, and $\langle x \rangle$ denotes the canonical prefix-free binary serialization of x; its length is $|\langle x \rangle|$. If A is an array on a finite ordered set Ω and $Q = (q_1, \ldots, q_r)$ is an index tuple, then $A[Q] = (A[q_1], \ldots, A[q_r])$, retaining order and multiplicity. If $Q \subseteq \Omega$ is a set, then $A[Q] = A|_Q$ uses the order induced by Ω.

Databases and computation. A database of length n is $DB_n = (b_0, \ldots, b_{n-1}) \in \mathbb{F}^n$, where $\mathbb{F} = \mathbb{F}_{2^B}$ is represented in a fixed polynomial basis. Thus field addition is bitwise XOR, denoted $\oplus$. Initialization installs DB_{n_0}, and after u one-block Update calls the current length is $n = n_0 + u \le N$. The quantity $M(B)$ is the bit complexity of one multiplication in $\mathbb{F}$. The commitment and Merkle-root length is λ_{vc} bits. A function is negligible if it is $\lambda^{-\omega(1)}$. All asymptotic conditions are for $\lambda \to \infty$, and "quasi-polynomial" means $2^{(\log \lambda)^{O(1)}}$ times a polynomial in the other displayed input sizes.

Partial words and MDS decoding. For a finite ordered set Ω, a partial word is a map $Y : \Omega \to \mathbb{F} \cup \{\perp\}$ with

$$\mathrm{supp}(Y) = \{q \in \Omega : Y[q] \neq \perp\}.$$

Relative to a word $C \in \mathbb{F}^\Omega$, its error count is

$$e_C(Y) = |\{q \in \mathrm{supp}(Y) : Y[q] \neq C[q]\}|.$$

An $[M, k]$ MDS code has the property that every k coordinates determine its message. We use the standard error-and-erasure criterion: if Y is a partial received word relative to a transmitted codeword C and $P = \mathrm{supp}(Y)$, then C is the unique codeword consistent with at most $e_C(Y)$ errors whenever

$$|P| - 2e_C(Y) \ge k. \tag{EE}$$

We also use that a nonzero polynomial of degree at most d over a field has at most d roots.

Typed objects and persistent random functions. A committed array is identified by its type γ, length m, and ordered contents A. The type includes the object's role, full interval tag, and all applicable representation, identity, and context fields. With public reference string *crs*, the commitment digest algorithm produces a λ_{vc}-bit root and opening state; an opening names an aligned coordinate set Q, the values $A[Q]$, and a proof. We assume correctness and binding against quasi-polynomial algorithms for this complete typed instance. In particular, except with the binding advantage, two accepted openings of one typed root cannot assign different values to the same coordinate.

Random oracles are persistent fixed random functions. Equivalently, in a lazy implementation each fresh typed input receives an independent uniform answer, and prefix-free domain tags name mutually independent families. Cache membership and query order are not visible to a machine. For a fixed typed audit context Ξ, let $\Gamma_{\mathcal{B}}$ be the space of complete configurations of a continuation $\mathcal{B}$, including its random tape, work tapes, and communication state, but not the ambient oracle tables. Restoring $\sigma \in \Gamma_{\mathcal{B}}$ restores only the machine: it neither erases nor reprograms any oracle. Once Ξ and the persistent functions are fixed, the transition maps of the continuation are deterministic.

Definition 2.1 (Distinct-identity receiver experiment and paired recovery)**.** *Fix a deterministic non-uniform polynomial-time adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, and let $t_{\mathcal{A}}(\lambda)$ be a polynomial upper bound on either phase and on each resumed continuation segment. In the first phase of the iDDA receiver experiment, the challenger and $\mathcal{A}_1$ execute the public setup, initialization, and adaptive Join and Update interactions of the scheme. At an adversarially chosen stopping length n, this phase fixes the original database DB^*, its public digest φ_n, the continuation state $st_{\mathcal{A}}$, and a transcript tr containing the challenger coins and messages needed to replay the phase. The second phase runs $\mathcal{B} = \mathcal{A}_2(st_{\mathcal{A}})$ through μ sequential Audit interactions at this same public state. We write*

$$(b, \varphi_n, DB^*, st_{\mathcal{A}}, tr) \leftarrow \mathsf{RecvExpt}^{\mathcal{A}}(1^{\lambda}, S, \mu)$$

for the resulting tuple.

In each Audit, the continuation fixes its identity and offline message before the online challenge. Conditional on those values, every randomly challenged active object receives an independent exact-uniform κ-tuple in its declared finite range; a full-opening object has a deterministic challenge. Later identities, messages, and checkpoints may depend on all earlier Audit interactions. A repeated identity makes the experiment reject, and $b = 1$ exactly on the event Pass *that all μ distinct-identity Audits accept.*

The recovery probability space first samples this experiment, including its setup coins, persistent random functions, and Audit challenges, then samples the finite uniform string ρ independently and runs the paired compressor and extractor displayed in Theorem 1.1. Thus every probability below is over these choices. The compressor may use tr to replay $\mathcal{A}_1$. The extractor receives neither tr nor DB^*, but only $\mathcal{B}$ and its displayed public inputs and advice. Both algorithms use the experiment's ambient public parameters and persistent functions. Inputs are paired *when the advice was produced by the compressor in this same public, continuation, oracle, and shared-coin context; transcript replay must reconstruct the corresponding digest, continuation state, and prior persistent-oracle context.*

Cost conventions. Publisher maintenance comprises construction and maintenance of encoded hierarchy objects, their Merkle trees, and every serialized public digest. In the simplified block-arithmetic model, one field operation and one hash on $O(B + \lambda_{\mathrm{vc}})$ input bits each cost $O(B + \lambda_{\mathrm{vc}})$ work; under $\lambda_{\mathrm{vc}} \le B$, displayed coding bounds use a factor B. Serialization is charged by its actual

bit length. In the genuine coding bit model, the field-operation factor is $M(B) + B$. Post-Init publisher maintenance excludes one-time Setup and Init and accounts for identity-dependent Join serving separately.

3 Overview

The proof has a publisher-maintenance part and a paired-recovery part. The first combines a prefix-separated binary-counter representation with the recursively mergeable MDS code AdditiveEC. The second uses the exact finite coin sampler TaggedChallengeSample, the accepting-spine rewind procedure AdaptiveRawExtract, and MDS error-and-erasure completion in TwoForestRecovery. The explicit local verified-extraction contract in Definition 4.7 supplies many usable occurrences; unconditional collision expansion converts them into enough distinct coordinates despite adaptive spine selection; and no-index MDS completion recovers each chunk. The local contract is a hypothesis, whereas this coupling, expansion, and completion chain is established in the proof. To obtain the strongest, variable-slack statement, we round the target slack down to a public dyadic value and set

$$\bar{\epsilon} = 2^{-\lceil \log_2(1/\epsilon) \rceil}, \qquad \eta = \frac{\bar{\epsilon}}{40}, \qquad \vartheta = 1 - 13\eta, \qquad R = 2^{\lceil \log_2(480\mathrm{e}/\bar{\epsilon}) \rceil}.$$

Thus $\epsilon/2 < \bar{\epsilon} \le \epsilon$ and $R = \Theta(1/\epsilon)$. The same rounding makes $\zeta_0 = \bar{\epsilon}/8$ use only $O(1 + \log(1/\epsilon))$ header bits, which the field-size condition lets the publisher bound absorb.

Separating the initialized prefix from later carries. At length $n = n_0 + u$, TwoForest represents the nonzero binary digits of n_0 and u as two independently tagged dyadic forests. The first forest partitions the initialized interval $[0, n_0)$ and is frozen after Init; the second starts empty and partitions the appended interval $[n_0, n_0 + u)$. Their $O(\log N)$ chunks, tagged by the forest, global starting point, and level, partition the database and are joined in chronological order by TwoForestJoin (Definition 4.2 and lemma 4.3). This separation addresses the publisher-side obstacle: in a single initialized binary counter, a post-Init carry can propagate through a large portion of the old prefix, so its cost cannot be bounded in terms of the number U of subsequent Updates alone. Here every such carry is confined to the initially empty suffix forest, with no condition relating n_0 and U.

The code makes each suffix carry cheap. Its recursive additive-polynomial construction, based on $\psi_t(X) = X^2 + c_t X$, evaluates on nested spaces V_t to give an $[R2^t, 2^t]$ MDS word, whose first 2^{t+1} coordinates also form a redundancy-two MDS word. Its essential identity combines two equal-size child words coordinatewise as

$$\mathsf{Merge}_t(C^{(0)}, C^{(1)})[z] = C^{(0)}[\psi_t(z)] + zC^{(1)}[\psi_t(z)],$$

so a size-2^t parent costs only $O(R2^t)$ field operations (Lemma 4.4). Every chunk caches its raw message, the redundancy-two restriction, and the full redundancy-R word, together with their typed trees. This is needed because the audit class of a persistent chunk can change as n changes; Update can then select a cached root instead of re-encoding that chunk.

If level ℓ is created during a suffix carry, it costs $O(BR2^\ell)$ in the simplified model and is created at most $\lfloor U/2^\ell \rfloor$ times in the first U Updates. Hence

$$\sum_{\ell=0}^{\lfloor \log_2 U \rfloor} O(BR2^\ell) \left\lfloor \frac{U}{2^\ell} \right\rfloor = O(BRU \log N) = O\left(\frac{BU \log N}{\epsilon} \right).$$

Serializing all U public digests adds $O(BU \log N + U|\langle\zeta_0\rangle|)$ work; the dyadic encoding and field-size condition absorb this term into the displayed bound. The implementation in TwoForest runs Setup, Init, Join, Update, and Audit on these objects: Setup prepares the code and typed namespaces, Init builds and freezes the prefix forest, Update changes only the suffix forest, Join materializes an identity's authenticated samples and labels, and Audit checks the typed offline and online openings. The same simplified node-independent post-Init metric treats one-time Setup and Init and identity-dependent Join serving separately. It gives $O(BN/\epsilon)$ publisher storage and replaces B by $M(B)+B$ for coding bit work; a full codeword tree has depth $O(\log N + \log(1/\epsilon))$. It is an aggregate, hence amortized, Update bound rather than a worst-case bound for an individual carry.

From accepted audits to candidate codeword symbols. For a sampled chunk c of length n_c, an identity stores

$$s_c = \left\lceil \max\left\{ \frac{S}{n} n_c, s_{\mathrm{lb}}(n) \right\} \right\rceil$$

positions sampled independently with replacement from its length-Rn_c word, where $s_{\mathrm{lb}}(n)$ is the public lower cutoff in Equation (6). The cutoff ensures that a sampled chunk has at least κ occurrences for the extraction estimates. Together, the proportional term, this cutoff, the recovery-validity bounds, and the $O(\log N)$-chunk decomposition keep total honest-node data at $O(S)$. Smaller mini chunks use their whole redundancy-two word, and tiny chunks use their raw word, avoiding a sampling argument in ranges too small to support it. For each identity, the sampled data are masked into the final quarter of a depth-robust-graph label array. Writing $\widetilde{h}_{c,i}$ for the final verified label array accumulated by the rewind procedure below, an accepted online opening of a terminal label and all its parents yields the candidate pair

$$\left(g_{c,id_i}(j),\ \widetilde{h}_{c,i}[q] \oplus H_c((\varphi_n, id_i), q, \widetilde{h}_{c,i}[\mathrm{Par}_c(q)]) \right), \qquad q = 3s_c + j.$$

Typed commitment binding makes accepted openings consistent at an opened coordinate, but the candidate need not be correct: usability is an analysis notion that the algorithm neither knows nor tests. The local extraction contract asserts that, on its good event, at least $(1-4\eta)s_c$ occurrences per selected sampled identity are usable; it gives the analogous redundancy-two and full-raw conclusions for mini and tiny chunks.

A transcript-free accepting spine. The recovery algorithms must agree on those selected identities and rewind points even though later audits may depend on earlier interactions and the extractor receives neither the receiver transcript nor the original database. TaggedChallengeSample first parses the finite shared uniform string ρ by bounded rejection. This is necessary because the challenge ranges need not have power-of-two size; conditional on the negligible parser-failure event not occurring, all named challenges are exactly uniform, mutually independent, and independent of the receiver experiment.

Given the public audit context, the continuation, the public allocation, and ρ, AdaptiveRawExtract tries K_{sp} complete audit paths driven by independent challenge tables, conditional on the fixed public context and persistent random functions. It selects the least path on which all μ distinct-identity audits accept and saves the machine configuration immediately after each identity submission and before its offline message. If a fresh path accepts with conditional probability p, its coupling to the received accepting execution is controlled by

$$p(1-p)^{K_{\mathrm{sp}}} \le \frac{1}{K_{\mathrm{sp}}+1}.$$

The choice of K_{sp} in Equation (16) makes this loss negligible while remaining quasi-polynomial. A checkpoint wrapper then repeatedly restores each saved machine configuration for the T_{ext} raw trials, without resetting the persistent random functions, and aggregates only values from trials whose offline and online predicates both accept. Its output is a partial candidate array for every sampled or mini chunk and a partial raw array for every tiny chunk. Once the supplied continuation state and public context are fixed, this computation uses no transcript. Its deterministic least-spine rule and last-write recurrence therefore give paired callers with the same persistent functions and ρ exactly the same arrays in quasi-polynomial time (Lemma 4.8).

Removing postselection bias. The central recovery difficulty remains that the identities on the accepting spine can themselves depend on answers of the shard random function G_c. One therefore cannot condition on spine success and then apply an ordinary independent occupancy bound to their sampled positions. The proof instead exposes the adaptive set of first arguments reached by the experiment, replay, and spine search, pads it to a deterministic size, and defers the local rewind queries until after this exposure. It can then apply the deletion-resilient collision lemma unconditionally. Since $R \geq 12\mathrm{e}/\eta$, retaining any $(1-4\eta)s_c$ usable occurrences from each of the selected r_c identities still leaves at least $(1-5\eta)r_c s_c$ distinct coordinates (Lemmas 4.9 and 4.10).

The public allocation is chosen to turn that expansion into precisely the decoding margin that is needed:

$$r_c = \min\left\{\mu, \left\lceil \frac{n_c}{\vartheta s_c} \right\rceil\right\}, \qquad c_c = \min\{n_c, \lceil \vartheta \mu s_c \rceil\}, \qquad d_c = n_c - c_c.$$

Let Y_c keep the first populated candidate at each coordinate, let P_c be its support, and let e_c count its unknown errors relative to the authentic codeword. An unusable occurrence can cause at most one bad first write. Combining the distinct-coordinate bound with the at most $4\eta r_c s_c$ unusable occurrences gives the pivotal estimate

$$|P_c| - 2e_c \geq (1-13\eta)r_c s_c = \vartheta r_c s_c,$$

and integrality together with the definition of r_c upgrades this to $|P_c| - 2e_c \geq c_c$. This is why collision expansion must survive both adaptive selection and deletion: support alone is insufficient when the extracted partial word can contain errors that the extractor cannot recognize.

No-index completion and final assembly. For every sampled chunk, both paired callers choose I_c to be the first d_c public evaluation points outside P_c. The compressor, which may use the transcript to recover the original database, sends only the authentic codeword values at I_c in canonical order. It sends neither I_c nor any support metadata. The transcript-free extractor recomputes the same set from the shared raw record, fills those values into Y_c, and obtains

$$(|P_c| + d_c) - 2e_c \geq c_c + (n_c - c_c) = n_c,$$

which is exactly the MDS error-and-erasure criterion. Mini chunks meet the same criterion in their $[2n_c, n_c]$ restriction, tiny chunks are recovered by their full raw openings, and TwoForestJoin reassembles the decoded messages in database order.

Finally, $s_c \geq (S/n)n_c$ and the definition of c_c imply

$$B \sum_{c \text{ sampled}} d_c \leq B \max\{n - (1-\epsilon)\mu S, 0\}.$$

The sampler, spine, local-extraction, binding, and unconditional-expansion losses are jointly negligible under the hypotheses of Theorem 1.1, which gives $\Pr[\mathsf{Pass} \wedge (DB_{\mathrm{ext}} \neq DB^*)] \leq \mathrm{negl}(\lambda)$

for the quasi-polynomial extractor. This conclusion requires the stated pairwise-distinct challenge identities and paired advice in the same public, continuation, persistent-oracle, and shared-coin context. It is a recoverability theorem; no replication probability conclusion is used or proved.

4 Proof of the Main Theorem

4.1 Parameters and the two-forest decomposition

We first isolate the public hypotheses. The local graph-extraction statement used later is recorded separately in Definition 4.7; it is a hypothesis of the recovery clause, not of publisher accounting.

Definition 4.1 (Admissible public parameters and recovery interface)**.** *Fix an absolute constant $c_\Delta > 0$. A publisher-admissible family consists, at each security parameter λ, of integers B, S, n_0, N, κ, Δ, a public integer $w = w(\lambda)$, a λ_{vc}-bit typed Merkle vector commitment, and the displayed values R, ζ_0 from Theorem 1.1. It satisfies*

$$2 \le \lambda \le n_0 < N \le \lambda^{O(1)}, \qquad 1 \le S \le n_0, \qquad 1 \le \kappa, w \le \lambda^{O(1)},$$

$$1 \le \Delta \le c_\Delta \log_2(2N), \qquad w \log_2 n_0 \ge 1, \qquad 2(\Delta+1)\kappa w \log_2 N \le S, \tag{1}$$

as well as $R\widehat{N} \le 2^B$, $1 \le \lambda_{\mathrm{vc}} \le B$, and $1 \le |\langle \zeta_0 \rangle| \le BS$. The public directed-graph interface, on a length $m \le 8N$, a domain tag, and a vertex $q \in [m]$, returns in polynomial time an increasing list of at most Δ parents, all smaller than q. For each tag these lists define the promised depth-robust graph. The graph description pp_{DRG} has $O(\lambda + \log N)$ bits. For purposes of the recovery theorem, the semantic content of this graph promise is exactly the uniform verified-extraction statement in Definition 4.7; publisher correctness uses only acyclicity and the displayed indegree and running-time bounds. The Merkle leaf hash binds a prefix-free encoding of the object type, full interval tag, array length, coordinate, and value. Public setup creates the field tables and independently domain-separated graph and random-oracle namespaces.

Call a publisher-admissible family recovery-admissible *if, in addition, $N, R, \kappa, \mu \le \lambda^{O(1)}$, every serialized digest has polynomial length, and the complete public recovery input has length $\mathrm{poly}(\lambda, t_{\mathcal{A}})$. Its public Audit interaction is represented by total deterministic transition maps* NextId, Offline, *and* Online *once the persistent random functions are fixed. A call, including parsing, communication, graph operations, oracle calls, and verification over all active chunks, has bit cost at most C_{tr}, where $L_{\varphi_n} := |\varphi_n|$ is the actual serialized current-digest length and*

$$C_{\mathrm{tr}} = t_{\mathcal{A}}(\lambda) + \mathrm{poly}(\lambda, N, R, B, \kappa, L_{\varphi_n}) \tag{2}$$

Malformed or overlong records have total rejecting branches. The typed vector commitment is correct and binding against quasi-polynomial algorithms, and all random-oracle families are persistent, cache hidden, and mutually domain separated. These input-size conditions ensure efficiency; they do not give either recovery algorithm additional information.

At length $n = n_0 + u$, the prefix and suffix binary expansions give the following tagged chunks. This dyadic organization is the binary-counter pattern behind static-to-dynamic transformations [BS80], applied separately on the two sides of the initialization boundary.

Definition 4.2 (Prefix-separated tagged chunks)**.** *For $0 \le u \le N - n_0$, let*

$$H(v) = \{\ell : \textit{bit } \ell \textit{ of } v \textit{ is } 1\}, \qquad \mathcal{K}(u) = \{(0,\ell) : \ell \in H(n_0)\} \cup \{(1,\ell) : \ell \in H(u)\}.$$

For $c = (f, \ell) \in \mathcal{K}(u)$ put $f_c = f$, $\ell_c = \ell$, $n_c = 2^{\ell_c}$, and

$$a_{(0,\ell)} = \sum_{\substack{j \in H(n_0) \\ j > \ell}} 2^j, \qquad a_{(1,\ell)} = n_0 + \sum_{\substack{j \in H(u) \\ j > \ell}} 2^j.$$

For $DB = (b_0, \ldots, b_{n-1})$, define

$$x_{n,c} = (b_{a_c}, \ldots, b_{a_c + n_c - 1}), \qquad \iota_{n,c} = (f, a_c, \ell, n_c). \tag{3}$$

Chunks are ordered first by f and then by decreasing ℓ. Denote the resulting split and concatenation maps by $\mathsf{TwoForestSplit}_{n_0,u}$ and $\mathsf{TwoForestJoin}_{n_0,u}$.

The binary intervals in this definition have the following elementary but essential reconstruction property.

Lemma 4.3 (Tagged chunks partition the database)**.** *For every $u \in [0, N - n_0]$,*

$$|\mathcal{K}(u)| \le 2(1 + \lfloor \log_2 N \rfloor), \qquad \sum_{c \in \mathcal{K}(u)} n_c = n_0 + u,$$

and

$$\mathsf{TwoForestJoin}_{n_0,u}\big(\mathsf{TwoForestSplit}_{n_0,u}(DB)\big) = DB.$$

Proof. Within either forest, the decreasing nonzero binary digits describe disjoint consecutive intervals whose lengths sum to the represented integer. The first forest partitions $[0, n_0)$ and the second partitions $[n_0, n_0 + u)$. Each binary expansion has at most $1 + \lfloor \log_2 N \rfloor$ nonzero digits. The three assertions follow. □

The next ingredient lets the binary carries update a codeword without re-encoding all of its descendants. It is a polynomial-evaluation MDS code in the Reed–Solomon tradition [RS60], arranged so that equal-size words admit a linear-work merge.

4.2 A recursively mergeable MDS code

Let $L_{\max} = \lceil \log_2 N \rceil$. Fix an ordered $\mathbb{F}_2$-basis of $\mathbb{F}$. Since $R\widehat{N} \le 2^B$ and R is a power of two, let $V_{L_{\max}} \subseteq \mathbb{F}$ be the span of the first $\log_2(R\widehat{N})$ basis vectors. For $t = L_{\max}, \ldots, 1$, take c_t to be the first nonzero vector in the current row-reduced basis of V_t and set

$$\psi_t(X) = X^2 + c_t X, \qquad V_{t-1} = \psi_t(V_t).$$

The restriction of ψ_t to V_t is linear with kernel $\{0, c_t\}$, so $|V_t| = R2^t$. Compute the canonical row-reduced basis of V_{t-1} and the canonical row-reduced linear section $\sigma_t : V_{t-1} \to V_t$. Order V_t by placing $\sigma_t(y), \sigma_t(y) + c_t$ consecutively for each point y in the fixed order of V_{t-1}, starting with lexicographic field-coordinate order on V_0. Write $z_{t,q}$ for the qth point.

For $x \in \mathbb{F}^{2^t}$ define a polynomial recursively by

$$P^{(0)}_{(x_0)}(X) = x_0, \qquad P^{(t)}_{x}(X) = P^{(t-1)}_{x^{(0)}}(\psi_t(X)) + X P^{(t-1)}_{x^{(1)}}(\psi_t(X)), \tag{4}$$

where $x = x^{(0)} \| x^{(1)}$. Define

$$\mathsf{Enc}_{2^t}(x)[z] = P^{(t)}_x(z) \quad (z \in V_t),$$

and let W_t be the first 2^{t+1} points of V_t. For arrays $C^{(0)}, C^{(1)} \in \mathbb{F}^{V_{t-1}}$, define

$$\mathsf{Merge}_t(C^{(0)}, C^{(1)})[z] = C^{(0)}[\psi_t(z)] + zC^{(1)}[\psi_t(z)]. \tag{5}$$

Public setup stores each z together with the position of $\psi_t(z)$ in V_{t-1}. These tables contain $O(R\widehat{N})$ field elements and indices.

AdditiveEC: Set up, encode, merge, or decode an additive-evaluation word.
Input: A mode $m \in \{\mathsf{setup}, \mathsf{encode}, \mathsf{merge}, \mathsf{decode}\}$ and a mode-specific payload: (B, N, R); (t, x); $(t, C^{(0)}, C^{(1)})$; or a partial word (t, Y), respectively. The public field representation and, outside setup mode, the tables above are ambient.
Output: Public encoding tables; $\mathsf{Enc}_{2^t}(x)$; $\mathsf{Merge}_t(C^{(0)}, C^{(1)})$; or the unique message within the stated error-and-erasure radius or $\perp$, respectively.
Guarantee: In decode mode, if $P = \{z : Y[z] \neq \perp\}$ and $Y|_P$ has e errors relative to a transmitted word, the transmitted message is returned whenever the criterion in Equation (EE) holds, namely $|P| - 2e \geq 2^t$.

1. In setup mode, construct the spaces, sections, orders, and lookup tables described above and output them.
2. In encode mode, recursively encode the two halves of x and combine them coordinatewise by Equation (5); at $t = 0$, evaluate the constant polynomial on V_0.
3. In merge mode, scan the level-t table and evaluate Equation (5) once at each output coordinate.
4. In decode mode, output $\perp$ if $|P| < 2^t$. Otherwise set $H = \lfloor (|P| - 2^t)/2 \rfloor$. For $h = 0, \ldots, H$, solve the Berlekamp–Welch equations $Q(z) = Y[z]E(z)$ on P, where E is monic of degree h and $\deg Q < 2^t + h$. Test every quotient Q/E of degree less than 2^t and apply the inverse of the fixed coefficient map $x \mapsto P_x^{(t)}$ to the unique candidate with at most H disagreements and output the resulting message; output $\perp$ if no such candidate exists.

The next lemma proves both the algebraic guarantees and the resource bounds of the displayed implementation.

Lemma 4.4 (Linear-work additive MDS coding)**.** *The map Enc_n, for $n = 2^t$, is an $[Rn, n]$ MDS code, and its restriction to W_t is an $[2n, n]$ MDS code. The algorithm* AdditiveEC *is correct. Setup uses $O(L_{\max} B^3)$ binary-linear-algebra work plus $O(R\widehat{N})$ field operations and table writes; Merge uses $O(Rn)$ field operations; Encode uses $O(Rn(1+\log n))$ field operations; and the displayed direct decoder uses at most $O((Rn)^4 + n^3)$ field operations.*

Proof. Induction in Equation (4) gives $\deg P_x^{(t)} < 2^t$. To prove injectivity, suppose the polynomial is zero and abbreviate its two child polynomials by A and D. The two preimages of $y \in V_{t-1}$ are z and $z + c_t$, and characteristic two gives

$$0 = P_x^{(t)}(z) + P_x^{(t)}(z + c_t) = c_t D(y).$$

Thus D vanishes at $|V_{t-1}| = R2^{t-1} \geq 2^{t-1}$ points while $\deg D < 2^{t-1}$, so $D = 0$; then $A = 0$. Induction recovers $x = 0$. Consequently the messages parameterize all polynomials of degree less than n, and any n distinct evaluations determine the message. This proves both MDS assertions.

Equation (4) is exactly the merge rule in Equation (5). One multiplication and one addition per coordinate give the merge bound, and $T(n) = 2T(n/2) + O(Rn)$ gives the encoding bound. If two degree-less-than-n polynomials each disagree with Y at at most $H = \lfloor (|P| - n)/2 \rfloor$ positions, they agree at at least n points and hence are equal. If the actual number of errors is e and $|P| - 2e \geq n$, the error locator of degree e appears in the search; the usual root count forces $Q = EP_x^{(t)}$, so division returns the transmitted polynomial. Direct Gaussian elimination gives the conservative decoding bound. Binary row reduction constructs every image and section, and the geometric sum of table sizes is $O(R\widehat{N})$, proving the setup bounds. □

We now use this code inside the exact tagged state maintained by the publisher.

4.3 The five-protocol two-forest construction

At a current length n, put $L(n) = \lfloor \log_2 n \rfloor$ and

$$s_{\mathrm{lb}}(n) = \left\lceil \frac{S 2^{L(n)}}{nw \log_2 n} \right\rceil. \tag{6}$$

A chunk is *tiny* if $n_c \leq \kappa$, *mini* if $\kappa < n_c \leq s_{\mathrm{lb}}(n)$, *small* if $s_{\mathrm{lb}}(n) < n_c \leq s_{\mathrm{lb}}(n)N/S$, and *large* otherwise. Small and large chunks are called *sampled*. For a sampled chunk set

$$s_c = \left\lceil \max\left\{ \frac{S}{n} n_c, s_{\mathrm{lb}}(n) \right\} \right\rceil, \qquad \tau_c = s_c, \qquad m_c = Rn_c. \tag{7}$$

For a mini chunk put $(\tau_c, m_c) = (2n_c, 2n_c)$, and for a tiny chunk put $m_c = n_c$. The recovery-validity inequalities imply, on sampled chunks,

$$1 \leq s_c \leq n_c, \qquad s_c \geq (S/n)n_c, \qquad s_c \geq \kappa. \tag{8}$$

Every chunk caches $x_{n,c}$, the restriction of $C_{n,c} = \mathsf{Enc}_{n_c}(x_{n,c})$ to W_{ℓ_c}, the full word $C_{n,c}$, and typed Merkle trees for all three arrays. Its active data word is

$$D_{n,c} = \begin{cases} x_{n,c}, & c \text{ tiny}, \\ C_{n,c}|_{W_{\ell_c}}, & c \text{ mini}, \\ C_{n,c}, & c \text{ sampled}. \end{cases} \tag{9}$$

Every tree, oracle, graph, and commitment namespace includes the full tag $\iota_{n,c}$ from Equation (3); this separates equal-size chunks in the two forests. Define the data-type tag

$$t_c^D = \begin{cases} \mathsf{raw}, & c \text{ tiny}, \\ \mathsf{mini}, & c \text{ mini}, \\ \mathsf{full}, & c \text{ sampled}, \end{cases} \qquad \gamma_c^D = (\mathsf{data}, \iota_{n,c}, t_c^D, m_c), \qquad \gamma_{c,id}^h = (\mathsf{label}, \iota_{n,c}, id, 4\tau_c). \tag{T1}$$

Let ser_{pp} be the finite prefix-free serialization of the additive-code tables, pp_{DRG}, the commitment reference string, the Merkle-hash identifier, the descriptions of the G, H, FS domain namespaces, and the scalar parameters. It contains neither random-oracle tables nor the identifier being defined. Let id_{pp} be the λ_{vc}-bit hash, under a dedicated setup separator, of ser_{pp}. The full setup remains ambient public data and is not copied into each digest. For $0 \leq \ell \leq L_{\max}$, define

$$\mathsf{slot}_{n,f,\ell} = \begin{cases} (\iota_{n,(f,\ell)}, \varphi_{n,(f,\ell)}), & (f,\ell) \in \mathcal{K}(u), \\ \perp, & (f,\ell) \notin \mathcal{K}(u), \end{cases}$$

where $\varphi_{n,c}$ is the typed Merkle root of $D_{n,c}$. The public digest is the canonical serialization

$$\varphi_n = \left\langle (\mathsf{slot}_{n,f,\ell})_{f\in\{0,1\},\, 0\le\ell\le L_{\max}}, (n, n_0, N, S, B, \kappa, R, w, \Delta, \zeta_0, id_{\mathrm{pp}}, (t_c^D)_{c\in\mathcal{K}(u)}) \right\rangle . \tag{T2}$$

Its parser recomputes $u = n - n_0$ and $\mathcal{K}(u)$, checks every active tag and root and every inactive $\perp$ slot, and verifies the scalar header and setup identifier. If $z_0 = |\langle\zeta_0\rangle|$, then

$$L_{\varphi_n} = O((1+\log N)(\lambda_{\mathrm{vc}} + \log N) + \log B + z_0). \tag{T3}$$

The typed vector commitment has exact interfaces

$$\mathsf{VC.Digest}(crs, \gamma, m, A) \to (\phi, aux), \qquad \mathsf{VC.Open}(crs, \gamma, m, aux, Q) \to \pi,$$

$$\mathsf{VC.Vf}(crs, \gamma, m, \phi, Q, A[Q], \pi) \to \{0,1\}. \tag{T4}$$

For every legal $\iota = (f, a, \ell, 2^\ell)$ and $1 \le \tau \le 2N$, Setup exposes independent persistent families

$$G_\iota : \{0,1\}^* \times \mathbb{N} \to [R2^\ell], \qquad H_{\iota,\tau} : \{0,1\}^* \to \mathbb{F}, \qquad FS_{\iota,\tau} : \{0,1\}^* \to [4\tau]^\kappa, \tag{T5}$$

and the parent map

$$\mathrm{Par}_{\iota,\tau}(q) = \mathsf{DRG.Parents}(pp_{\mathrm{DRG}}, 1^\lambda, 4\tau, \zeta_0, (\iota,\tau), q). \tag{T6}$$

Thus $G_c = G_{\iota_{n,c}}$, while H_c, FS_c, and Par_c are indexed by the pair $(\iota_{n,c}, \tau_c)$. This second component is mandatory because the active sampling length of a persistent chunk may change. For sampled c, put $g_{c,id}(j) = G_c(id, j)$ for $j \in [s_c]$; these values are independent uniform coordinates sampled with replacement. On a mini chunk $g_{c,id}(j) = j$ for $j \in [2n_c]$.

Put $\mathcal{Q}_c = \{3\tau_c, \dots, 4\tau_c - 1\}$. The node stores $d_{c,id}[j] = D_{n,c}[g_{c,id}(j)]$ and the label array

$$h_{c,id}[q] = \begin{cases} H_c((\varphi_n, id), q, h_{c,id}[\mathrm{Par}_c(q)]), & q < 3\tau_c, \\ H_c((\varphi_n, id), q, h_{c,id}[\mathrm{Par}_c(q)]) \oplus d_{c,id}[q - 3\tau_c], & q \in \mathcal{Q}_c. \end{cases} \tag{T7}$$

Here the parent tuple is ordered and has size at most Δ. Let $(\chi_{c,id}, aux^h_{c,id})$ be the result of committing to this complete label array under $\gamma^h_{c,id}$. For any finite tuple $Q = (q_1, \dots, q_r) \in [4\tau_c]^r$, put

$$U_c(Q) = \bigcup_{j=1}^{r} (\{q_j\} \cup \mathrm{Par}_c(q_j)). \tag{T8}$$

The offline predicate is as follows. Define

$$F_c = FS_c(\chi_{c,id}, id), \qquad J_c = \{q - 3\tau_c : q \in F_c \cap \mathcal{Q}_c\}, \qquad K_c = \{g_{c,id}(j) : j \in J_c\}. \tag{T9}$$

The prover supplies $h^0_c[U_c(F_c)]$, occurrence-indexed values $d^0_c[J_c]$, and separate label and data proofs. The verifier first checks that occurrences mapped to one coordinate have equal values and forms the coordinate-indexed array $\overline{d}^0_c[K_c]$. It then verifies

$$\mathsf{VC.Vf}(crs, \gamma^h_{c,id}, 4\tau_c, \chi_{c,id}, U_c(F_c), h^0_c[U_c(F_c)], \pi^{0,h}_c) = 1,$$

$$\mathsf{VC.Vf}(crs, \gamma^D_c, m_c, \varphi_{n,c}, K_c, \overline{d}^0_c[K_c], \pi^{0,D}_c) = 1, \tag{T10}$$

and checks Equation (T7) at every $q \in F_c$, using the supplied occurrence value when $q \in \mathcal{Q}_c$.

After all offline predicates accept, the online verifier independently samples $Q_c \in \mathcal{Q}_c^\kappa$, receives $h_c^e[U_c(Q_c)]$ and π_c^e, and checks exactly

$$\mathsf{VC.Vf}(crs, \gamma_{c,id}^h, 4\tau_c, \chi_{c,id}, U_c(Q_c), h_c^e[U_c(Q_c)], \pi_c^e) = 1. \tag{T11}$$

A tiny chunk instead opens its complete raw word under γ_c^D. Every verification passes the tag, length, aligned index set, values, and proof explicitly. All chunks run in parallel, with at most $s_{\mathrm{lb}}(n)$ sequential label-oracle dependency rounds from receipt of the identity through the online response.

For later rewinding, let Ξ denote the typed public audit context specified above together with the fixed persistent functions, and use the configuration space $\Gamma_{\mathcal{B}}$ from Section 2. Let $\mathsf{Stat} = \{\mathsf{accept}, \mathsf{reject}, \mathsf{halt}\}$ and $\mathsf{Id} = \{0,1\}^*$. Let $\mathfrak{O}_{\mathrm{off}}$ and $\mathfrak{O}_{\mathrm{on}}$ be the finite spaces of the typed records specified in Equations (T9) to (T11). For an active chunk, set $\mathcal{C}_c = \mathcal{Q}_c^\kappa$ when it is sampled or mini and $\mathcal{C}_c = \{[n_c]\}$ when it is tiny. With Com_c denoting the label-commitment space, the transition interfaces are

$$\mathsf{NextId}_\Xi : \Gamma_{\mathcal{B}} \to (\mathsf{Id} \times \Gamma_{\mathcal{B}}) \cup (\{\mathsf{halt}\} \times \Gamma_{\mathcal{B}}),$$

$$\mathsf{Offline}_\Xi : \Gamma_{\mathcal{B}} \times \mathsf{Id} \to \mathsf{Stat} \times \Gamma_{\mathcal{B}} \times (\mathfrak{O}_{\mathrm{off}} \cup \{\bot\}),$$

$$\mathsf{Online}_\Xi : \Gamma_{\mathcal{B}} \times \mathsf{Id} \times \prod_{c \text{ active}} \mathcal{C}_c \times \prod_{c \text{ sampled or mini}} \mathsf{Com}_c \to \mathsf{Stat} \times \Gamma_{\mathcal{B}} \times (\mathfrak{O}_{\mathrm{on}} \cup \{\bot\}). \tag{T12}$$

The first map runs until the next identity submission or halt and returns the exact configuration immediately after submission and before any offline message. The second executes all predicates in Equations (T9) and (T10); it returns the verified record, including every $\chi_{c,id}$, exactly on acceptance. The third sends the explicit challenges, executes Equation (T11) and the tiny predicates, and returns the exact post-interaction configuration and every received record on a nonhalting branch. Rejection, halt, malformed, and missing-record branches return their named status and $\bot$ record as appropriate. For fixed random functions all three maps are deterministic, and each has cost at most C_{tr} from Equation (2).

The publisher state is $(n_0, u, \mathcal{F}_0, \mathcal{F}_1)$. Forest $\mathcal{F}_0$ represents the fixed initial sequence with origin 0, and $\mathcal{F}_1$ represents the suffix with origin n_0. Within a forest of local length v, a binary-carry level ℓ has local start $\sum_{j \in H(v), j > \ell} 2^j$ and the corresponding full global interval tag.

TwoForest: The tagged Setup–Init–Join–Update–Audit protocol suite.
Input: Setup receives 1^λ and a publisher-admissible public parameter tuple from Definition 4.1; Init receives (S, DB_{n_0}); Join receives a valid publisher state, its digest, and id; Update receives a valid state, one block b_n, and a participant set; Audit receives a digest, an identity, and a purported node state.
Output: Respectively public parameters; the initial publisher state and digest; a current node state; the updated publisher state, digest, and refreshed honest-node states, with the publisher state maintained in place; or accept/reject.
Invariant: Each forest is the tagged dyadic decomposition of its own sequence, and every active chunk stores the three code representations and typed trees defined above.

1. Setup: invoke AdditiveEC(setup, (B, N, R)), initialize the typed Merkle commitment, and publish the independently tagged shard, label, Fiat–Shamir, and graph namespaces.

2. Init: starting with empty $\mathcal{F}_0$, append $b_0, \ldots, b_{n_0-1}$. For each append, create the three tagged level-zero objects, merge through the trailing occupied levels using AdditiveEC(merge, $(t, C^{(0)}, C^{(1)})$), and build the three tagged parent trees. Freeze the resulting $\mathcal{F}_0$, set $\mathcal{F}_1 = \varnothing$, select the active representations in Equation (9), and serialize φ_{n_0}.

3. Join: parse the digest and recompute all chunks and classes. Send authenticated data symbols for each sampled or mini shard and the full raw word for each tiny chunk. Verify those openings, compute and commit the labels in Equation (T7), and store the complete label arrays, required data, authentication paths, and offline records.

4. Update: append b_n only to $\mathcal{F}_1$. Create its tagged singleton; while the old suffix length has a trailing occupied level, merge that older level with the newer carry, build the three parent trees, and release the children. Leave $\mathcal{F}_0$ unchanged, increment u, select the cached current representations, serialize φ_{n+1}, and invoke Join independently for each displayed participant.

5. Audit: verify the public header and offline records, sample the independent tuples Q_c, verify all openings on Equation (T8) and every tiny raw opening, and accept exactly when all typed predicates and the common response-round bound hold.

Correctness of the five entry points follows from the common forest invariant; the same invariant also exposes the carry cost.

Lemma 4.5 (Two-forest protocol correctness and publisher accounting)**.** *Under publisher admissibility, TwoForest is honestly correct. Init uses $O(BRn_0 \log N)$ simplified work. If the old suffix length u has q trailing one-bits, one node-independent Update uses*

$$O(BR2^q + B \log N + L_{\varphi_{n+1}}) \tag{13}$$

simplified work. For every $U \in [1, N - n_0]$, the first U Updates use

$$O(BRU \log N + Uz_0) \tag{14}$$

simplified work, or $O((M(B) + B)RU \log N + Uz_0)$ bit work. Publisher storage is $O(BRN)$, a full tree has depth $O(\log N + \log R)$, and honest node state is $O(BS)$ under the block-size hypothesis in Theorem 1.1. The bounds include each serialized digest and exclude Setup, Init, and identity-dependent Join output from the post-Init maintenance measure. More precisely, one honest Join at

length n materializes

$$L_{\mathrm{J}}(n, id) = O(BS + \lambda_{\mathrm{vc}} S(\log N + \log R) + L_{\varphi_n}) \tag{U1}$$

bits of transcript and node state. Thus d participants contribute $O(dBS)$ bits under that block-size hypothesis, and the single public digest makes the complete externally materialized output $O((d+1)BS)$. The mutable publisher hierarchy is maintained in place and is not reserialized in this output quantity.

Proof. Consider one forest with origin o. A singleton starts at $o + v$. If the old local length has q trailing ones, the level-j message is older than and adjacent to the current carry for every $j < q$. The rule in Equation (5) therefore produces their chronological concatenation. The parent begins at the old child's global start, so induction gives exactly the interval tag in Equation (3). Higher levels remain unchanged. This proves the invariant. It also proves the per-call coding bound because the geometric sum of created word and tree sizes is $O(R2^q)$.

For compatibility with the new-before-old merge convention in the comparison iDDA construction [SSM26], no cached word must be changed. If Rev_m reverses a word and $\widetilde{\mathsf{Enc}}_m(y) = \mathsf{Enc}_m(\mathsf{Rev}_m(y))$, then

$$\mathsf{Rev}_{2m}(u \| v) = \mathsf{Rev}_m(v) \| \mathsf{Rev}_m(u).$$

Thus a protocol-facing new-before-old merge is conjugate to the chronological old-before-new merge used here. The same reversal applies to raw tiny chunks.

The initial forest is built once and is never touched by Update. During U suffix appends, a suffix chunk of size 2^ℓ is created $\lfloor U/2^\ell \rfloor$ times. Hence

$$\sum_{\ell=0}^{\lfloor \log_2 U \rfloor} O(BR2^\ell) \left\lfloor \frac{U}{2^\ell} \right\rfloor = O(BRU(1 + \log U)) = O(BRU \log N).$$

By Equation (T3), $R\widehat{N} \le 2^B$, and $\lambda_{\mathrm{vc}} \le B$, serializing all U digests costs $O(BU \log N + Uz_0)$, which proves Equation (14). Replacing the field-operation factor B by $M(B) + B$ gives the bit-work bound. The same level count applied to the n_0 Init appends gives $O(BRn_0 \log N)$.

The active full words contain $R \sum_{c \in \mathcal{K}(u)} n_c = Rn \le RN$ field elements. The raw words, redundancy-two restrictions, typed trees, and a constant number of carry buffers change this by only a constant factor. This proves storage, and the largest tree has at most RN leaves.

For honest-node space, Equation (7) and Lemma 4.3 give

$$\sum_{c \text{ sampled}} s_c \le S + O(|\mathcal{K}(u)|(s_{\mathrm{lb}}(n) + 1)) = O(S),$$

using Equation (1), $n \ge n_0 \ge \lambda$, and $N \le \lambda^{O(1)}$. In each forest the total size of mini chunks is less than $2s_{\mathrm{lb}}(n)$ and the total size of tiny chunks is less than 2κ, so the two forests still contribute $O(S)$ block values. Merkle paths use $O(\lambda_{\mathrm{vc}} S(\log N + \log R)) = O(BS)$ bits. The label recurrence and typed commitment correctness make every honest offline and online opening verify. Finally, all three representations already exist, so a class change only selects another cached root; its participant-specific refresh is precisely the separately charged Join call. This completes the correctness and resource proof. The same count includes every value and authentication path written by Join and the L_{φ_n}-bit public header, which proves Equation (U1); summing it over d independent sessions gives the stated output bounds. □

The remainder of the proof establishes the paired recovery clause.

4.4 Exact challenges and accepting-spine extraction

Fix a recovery-valid length $n = n_0 + u$, meaning that the inequalities in Equation (1) hold. For a sampled chunk define the public allocation

$$r_c = \min\left\{\mu, \left\lceil \frac{n_c}{\vartheta s_c} \right\rceil\right\}, \qquad c_c = \min\{n_c, \lceil \vartheta \mu s_c \rceil\}, \qquad d_c = n_c - c_c. \tag{15}$$

For mini and tiny chunks put $(r_c, c_c, d_c) = (1, n_c, 0)$. This allocation is a function only of public data. Let $r_{\max} = \max(\{1\} \cup \{r_c : c \text{ sampled}\})$.

Set $T_{\text{ext}} = \lceil \lambda^{\log_2 \lambda} \rceil$. With $L = \lfloor \log_2 \lambda \rfloor$, define

$$a_\epsilon = \lfloor \eta \kappa L \rfloor, \quad e_\epsilon = \max\left\{1, \left\lfloor \frac{1}{4} \min\{\lfloor \sqrt{a_\epsilon} \rfloor, L^2\} \right\rfloor\right\}, \quad K_{\text{sp}} = 2^{e_\epsilon}. \tag{16}$$

For each sampled or mini chunk let $b_c = \max\{1, \lceil \log_2 \tau_c \rceil\}$. There are two disjoint public tag sets. The spine set contains $(\mathsf{spine}, a, i, c, v)$ for $a \in [K_{\text{sp}}]$, $i \in [\mu]$, and $v \in [\kappa]$, with c ranging over all sampled or mini chunks. The raw set contains trial tags $(\mathsf{trial}, i, t, c, v)$ for $i \in [r_{\max}]$, $t \in \{1, \ldots, T_{\text{ext}}\}$, and canonical tags $(\mathsf{canonical}, i, c, v)$, again with c sampled or mini and $v \in [\kappa]$. Tiny-chunk challenges are deterministic full openings.

For either finite tag set $\mathcal{I}_X$, put

$$A_X = \lambda + \lceil \log_2 \max\{1, |\mathcal{I}_X|\} \rceil + 2.$$

For a tag belonging to chunk c, allocate A_X disjoint b_c-bit blocks. Interpret a block as an integer in $[2^{b_c}]$ and select the first block below τ_c; fail if none exists. Adding $3\tau_c$ converts the selected integer to an element of $\mathcal{Q}_c$. With $c(\alpha)$ denoting the chunk in tag α, the exact substring lengths are

$$D_X = A_X \sum_{\alpha \in \mathcal{I}_X} b_{c(\alpha)}, \qquad \rho_{\text{sp}} \in \{0,1\}^{D_{\text{sp}}}, \qquad \rho_{\text{raw}} \in \{0,1\}^{D_{\text{raw}}}, \tag{S1}$$

and the shared string is $\rho = \rho_{\text{sp}} \| \rho_{\text{raw}}$ of length $D_{\text{sp}} + D_{\text{raw}}$.

TaggedChallengeSample: Parse finite shared coins into exact independent audit challenges.
Input: A public sampling state $\mathfrak{X}$ containing the tagged chunks, $(\tau_c)_c$, κ, μ, $r_{\max}$, K_{sp}, and T_{ext}, and a bit string $\rho = \rho_{\text{sp}} \| \rho_{\text{raw}}$ of the exact length in Equation (S1).
Output: The complete spine, raw-trial, and canonical challenge tables, or $\mathsf{fail}_{\text{sp}}$ or $\mathsf{fail}_{\text{raw}}$.
Guarantee: Conditional on neither failure, every selected coordinate is exactly uniform in its chunk range, all coordinates are mutually independent, and they are independent of the receiver experiment.

1. In lexicographic tag order, partition ρ_{sp} and ρ_{raw} into the prescribed disjoint blocks; reject a string of any other length.
2. For each spine tag, set a spine-failure flag if none of its A_{sp} blocks is below τ_c; otherwise record the first such value.
3. Independently apply the same rule with A_{raw} to every raw and canonical tag and set a raw-failure flag when necessary. Thus both flags are defined on every coin-string outcome.
4. If a flag is set, return its named failure value, giving spine failure priority only for the output syntax. Otherwise add $3\tau_c$ to each recorded value and output the named tables.

The bounded-rejection construction has an exact distribution, as recorded next.

Lemma 4.6 (Exact bounded-rejection challenge sampling)**.** *For uniform correctly sized ρ, each of the two failure events in* TaggedChallengeSample *has probability at most $2^{-\lambda-2}$. Conditional on success, its distribution has the stated exact product form. Its running time is linear in its coin-string length, and that length is quasi-polynomial under the parameter conditions of Theorem 1.1.*

Proof. Because $2^{b_c} < 2\tau_c$ unless $\tau_c = 1$, a single block succeeds with probability at least $1/2$. One tag therefore fails with probability at most 2^{-A_X}, and a union bound over $|\mathcal{I}_X|$ tags is at most $2^{-\lambda-2}$. Conditional on success for one tag, its first accepted block is uniform on $[\tau_c]$. Different tags use disjoint bits, and their success events also depend on disjoint bits, so conditioning on global success preserves independence. The coin string is independent of the experiment. Finally, $\tau_c \le 2N$, while the two tag sets have quasi-polynomial size once K_{sp} and T_{ext} do; scanning every allocated block is linear in their total length. □

We next state precisely the local graph-extraction property used by the rewind procedure. It isolates the verified-occurrence conclusion supplied by the iDDA extraction analysis [SSM26] for the exact typed Audit relation required here. The chosen graph and commitment instantiation must satisfy this explicit hypothesis; the transcript-free coupling and collision arguments that follow are proved locally.

Definition 4.7 (Local verified-extraction contract)**.** *Fix the typed audit relation in Equations (T7) and (T8). Starting at a saved configuration immediately after an identity is submitted and before its offline message, a raw trial restores that machine configuration, retains all persistent random functions, executes the offline transition, and, if it accepts, executes the online transition with an independent challenge. Only label values in an accepted online typed opening, and raw values in an accepted tiny-chunk online opening, are written. Offline Fiat–Shamir openings are checked but never enter this aggregate. For each coordinate, later accepted online writes replace earlier writes.*

More explicitly, let $b^{\mathrm{off}}_{i,t}$ and $b^{\mathrm{on}}_{i,t}$ be the two trial statuses, let $Q^{e}_{c,i,t}$ be its named online challenge, and let $h^{e}_{c,i,t}$ be the received online label array. Define

$$\mathsf{W}_{c,i,t}[q] = \begin{cases} h^{e}_{c,i,t}[q], & b^{\mathrm{off}}_{i,t} = b^{\mathrm{on}}_{i,t} = \mathsf{accept} \text{ and } q \in U_c(Q^{e}_{c,i,t}), \\ \bot, & \text{otherwise}, \end{cases}$$

and, coordinatewise,

$$\mathsf{H}^{(0)}_{c,i}[q] = \bot, \qquad \mathsf{H}^{(t)}_{c,i}[q] = \begin{cases} \mathsf{W}_{c,i,t}[q], & \mathsf{W}_{c,i,t}[q] \neq \bot, \\ \mathsf{H}^{(t-1)}_{c,i}[q], & \mathsf{W}_{c,i,t}[q] = \bot. \end{cases} \tag{R1}$$

The final verified label array is $\widetilde{h}_{c,i} = \mathsf{H}^{(T_{\mathrm{ext}})}_{c,i}$. On a tiny chunk, define $\mathsf{W}^{\mathrm{tiny}}_{c,t}[q]$ to be the received raw value exactly when both trial statuses accept, and $\bot$ otherwise; applying the same recurrence gives its final partial raw array. Thus every offline rejection, online rejection, halt, or missing record makes no write.

For a sampled chunk, selected identity id_i, and occurrence $j \in [s_c]$, put $q = 3s_c + j$. If the final verified label array contains q and all its parents, its candidate is

$$\Big(g_{c,id_i}(j), \widetilde{h}_{c,i}[q] \oplus H_c((\varphi_n, id_i), q, \widetilde{h}_{c,i}[\mathrm{Par}_c(q)])\Big); \tag{17}$$

otherwise it is $\perp$*. A candidate is* usable *if its value equals the committed codeword at its displayed coordinate. The procedure never tests usability. The mini-chunk definition is identical with the injective map* $j \mapsto j$ *on* $[2n_c]$*, and a tiny chunk records the latest accepted full raw opening.*

The local verified-extraction contract asserts the following, with its internal extraction slack instantiated as 2η*. On the joint probability space of the receiver experiment, persistent random functions, and shared coins, there is an event* $\mathsf{Good}_{\mathrm{loc}}$ *contained in the event that both coin parsers succeed and an accepting spine is found, such that*

$$\Pr[\mathsf{Pass} \wedge \neg \mathsf{Good}_{\mathrm{loc}}] \le 2^{-\lambda-1} + \frac{1}{K_{\mathrm{sp}}+1} + K_{\mathrm{sp}} q_{\mathrm{loc}} \left(\frac{8N}{T_{\mathrm{ext}}+1} + (1-\eta)^{\kappa} \right) + \frac{K_{\mathrm{sp}}\mu}{T_{\mathrm{ext}}+1} + \nu_{\mathrm{bind}}(\lambda), \quad (18)$$

and, on $\mathsf{Good}_{\mathrm{loc}}$*, every sampled chunk and every selected identity* $i < r_c$ *has at least* $(1-4\eta)s_c$ *usable occurrences; each mini chunk has at least* $(1-4\eta)2n_c$ *usable, distinct-coordinate occurrences for identity zero; and each tiny chunk is recovered completely and correctly. Here*

$$q_{\mathrm{loc}} = 12\mu(1 + \lfloor \log_2 N \rfloor)(C_{\mathrm{tr}} + N + \kappa + 1),$$

and ν_{bind} *is the advantage of one quasi-polynomial commitment binding adversary. This assertion is required uniformly for the exact transitions in Equations (T10) and (T11), the sampled word* $\mathsf{Enc}_{n_c}(x_{n,c})$*, its* W_{ℓ_c} *mini restriction, the full-opening tiny word* $x_{n,c}$*, and every full instance and length tag* $(\iota_{n,c}, \tau_c)$*. Together with the typed commitment and parent interfaces, these are the complete compatibility premises of the contract.*

An accepting spine is a list $\mathcal{S} = ((id_i, \sigma_i))_{i\in[\mu]}$ obtained on one execution in which all μ audits accept, the identities are distinct, and σ_i is the complete configuration immediately after submission of id_i and before its offline message. From $\mathcal{S}$, define a checkpoint wrapper that returns these saved pairs in order, delegates each offline or online transition to the original continuation from the saved configuration, and advances exactly once after every local audit branch. Its next saved checkpoint is therefore independent of audit history supplied to the wrapper.

AdaptiveRawExtract: Find an accepting adaptive path and extract verified candidate symbols at its saved checkpoints.
Input: A public extraction state $\mathfrak{X}$ containing the typed audit context, deterministic continuation $\mathcal{B}$, allocation $(r_c)_c$, K_{sp}, and T_{ext}, and the shared coin string ρ.
Output: Candidate pairs or $\perp$ for each sampled (c,i,j) with $i < r_c, j < s_c$, one length-$2n_c$ candidate array per mini chunk, and one partial raw array per tiny chunk.
Invariant: Machine configurations may be restored, but the same fixed random functions and their lazy tables persist across every search and rewind.

1. Invoke TaggedChallengeSample$(\mathfrak{X}, \rho)$. On either sampler failure, return the correctly shaped all-$\perp$ record.

2. For $a = 0, \ldots, K_{\mathrm{sp}} - 1$, restore the initial configuration of $\mathcal{B}$ and execute μ sequential audits with spine table a. Reject that path on a halt, a repeated identity, or any failed offline or online transition. Save every pre-offline pair $(id_{a,i}, \sigma_{a,i})$ on a fully accepting path and select the least accepting a. Return the all-$\perp$ record if none exists.

3. Form the checkpoint wrapper from the selected spine. At its ith saved checkpoint, perform the T_{ext} raw trials in increasing trial order, always restoring σ_i and retaining the oracle tables. If an offline transition rejects or halts, do not invoke its online transition and make no write; if an online transition rejects or halts, likewise make no write. Aggregate only accepted online verified label and tiny-word writes by Equation (R1). Mini and tiny chunks use only checkpoint $i = 0$; later checkpoints make no assignment to those outputs.

4. Form sampled and mini candidates by Equation (17). After the trials at checkpoint i, use its separately tagged canonical challenge in one delegated audit so that the wrapper advances to checkpoint $i+1$. A canonical offline rejection or halt advances immediately; after a canonical offline acceptance, the online branch advances regardless of its terminal status. Return all candidate and tiny arrays after $r_{\max}$ checkpoints.

We now justify the accepting-path coupling, shared-output property, and total cost of this extraction procedure.

Lemma 4.8 (Transcript-free accepting-spine extraction)**.** *Under the recovery conditions of Theorem 1.1, the following hold.*

(i) Jointly with Pass*, the probability that no accepting spine is found is at most* $2^{-\lambda-1} + 1/(K_{\mathrm{sp}} + 1)$.

(ii) Two callers with the same public context, continuation state, persistent functions, and ρ *select the same spine and obtain the same output from* AdaptiveRawExtract*. This computation uses no experiment transcript once the continuation state is fixed.*

(iii) The failure probability of the usable-occurrence conclusions in Definition 4.7, together with the two sampler failures and the spine miss, is negligible.

(iv) The algorithm runs in time quasi-polynomial in $(\lambda, t_{\mathcal{A}})$.

Proof. Condition on the public context, the initial continuation configuration, and the complete persistent random functions, and then on successful parsing of both independent coin substrings.

Cache membership is hidden, so a full path is a deterministic function of its fresh challenge table. If one such table produces μ accepting distinct-identity audits with probability p, then the received experiment table and the K_{sp} search tables are independent samples from the same conditional distribution. Therefore

$$\Pr[\mathsf{Pass} \wedge \text{no searched path accepts}] = p(1-p)^{K_{\mathrm{sp}}} \le \frac{1}{K_{\mathrm{sp}}+1}.$$

Averaging and adding both sampler failures proves (i).

All transitions and parsers are deterministic after fixing the displayed data. Both callers consequently choose the same least accepting path. The checkpoint wrapper invokes the original continuation at precisely the saved configurations and makes those checkpoints audit-history independent. The last-write recurrence and candidate map are deterministic, proving (ii).

It remains to check the variable-slack balance. Put $y = \eta\kappa/L$. Since $\bar{\epsilon} > \epsilon/2$ and $\epsilon\kappa = \omega(\log\lambda)$, one has $y \to \infty$. From Equation (16),

$$e_\epsilon = \frac{L}{4}\min\{\sqrt{y}, L\} + O(1) = \omega(L), \qquad e_\epsilon = o(\eta\kappa), \qquad e_\epsilon \le L^2/4 + 1. \tag{19}$$

Hence K_{sp}^{-1} is negligible, $\log K_{\mathrm{sp}} = o(\eta\kappa)$, and $K_{\mathrm{sp}}\mu/(T_{\mathrm{ext}}+1)$ is negligible. Moreover $(1-\eta)^\kappa \le \exp(-\eta\kappa)$, while q_{loc} is polynomial. Thus every term in Equation (18) is negligible; ν_{bind} is negligible by quasi-polynomial binding. With Lemma 4.6, this proves (iii).

Finally, $K_{\mathrm{sp}}, T_{\mathrm{ext}} \le 2^{O((\log\lambda)^2)}$, and all transition and record sizes are polynomial in the displayed parameters. The number of transitions is $O(K_{\mathrm{sp}}\mu + \mu T_{\mathrm{ext}})$, so the claimed running time follows from Equation (2). □

The selected identities depend on oracle answers. We therefore need an expansion estimate that remains valid after deletions and an exposure argument that applies it without conditioning on spine success.

4.5 Collision expansion and completion

Lemma 4.9 (Deletion-resilient adaptive shard expansion)**.** *Let $0 < \eta \le 1/40$, $\vartheta = 1 - 13\eta$, $1 \le s \le m$, and $R \ge 12\mathrm{e}/\eta$. Let an adaptive process make at most T distinct queries to a random function $G : \{0,1\}^* \to [Rm]^s$ whose fresh answer is an independent uniform s-tuple. After exposure, retain an arbitrary set $A_x \subseteq [s]$ of size at least $(1-4\eta)s$ for each queried input x. Put $r_* = \lceil m/(\vartheta s)\rceil$. Except with probability at most*

$$\sum_{r=1}^{\min\{T, r_*\}} \binom{T}{r} 4^{-\eta s r}, \tag{20}$$

every J of queried inputs satisfies

$$\left|\bigcup_{x\in J}\{G(x)[j] : j \in A_x\}\right| \ge \min\{m, (1-5\eta)s|J|\}. \tag{21}$$

If $\eta s = \omega(\log\lambda)$ and $\log_2\max\{2,T\} = o(\eta s)$, the failure probability is negligible.

Proof. Pad an early-stopping process with fresh dummy inputs. Fix $r \le r_*$ query positions and flatten their answers into $q = sr$ uniform draws from $[Rm]$. Since $\vartheta > 1/2$ and $s \le m$, one has

$q < 3m$. Conditional on all prior draws, the probability that the next draw repeats is less than $3/R$. If at least ηq draws repeat, some $d = \lceil \eta q \rceil$ positions witness this. Iterated conditioning and a union bound give

$$\Pr[\text{at least } \eta q \text{ repeats}] \le \binom{q}{d}(3/R)^d \le \left(\frac{3\mathrm{e}}{R\eta}\right)^d \le 4^{-\eta q}.$$

Union-bounding over the $\binom{T}{r}$ choices and all $r \le r_*$ proves Equation (20). Deleting occurrences cannot increase their inherited repeat count. Thus at least $(1-4\eta)q$ retained occurrences lose fewer than ηq new coordinates and occupy at least $(1-5\eta)q$ coordinates. For $|J| > r_*$, choose an r_*-subset and use

$$(1-5\eta)sr_* \ge \frac{1-5\eta}{1-13\eta}m \ge m.$$

This proves Equation (21). Finally, with $z = T4^{-\eta s}$, the sum in Equation (20) is at most $z/(1-z)$; its logarithm is $-(2-o(1))\eta s = -\omega(\log \lambda)$. □

The collision lemma must be invoked on a total exposure process, rather than after selecting the potentially shard-oracle-dependent spine.

Lemma 4.10 (Unconditional expansion for the selected spine)**.** *For every sampled chunk c, there is a padded adaptive exposure process whose query set $\mathcal{X}_c^{\mathrm{pad}}$ contains exactly p_{sel} distinct inputs to the vector-valued shard random function. Write* SampleOK *for success of both coin parsers and* SpineOK *for success of the accepting-spine search. On* SampleOK $\cap$ SpineOK*, the set contains every selected identity $id_0, \dots, id_{r_c-1}$, and it satisfies*

$$\log_2 p_{\mathrm{sel}} = o(\eta s_c). \tag{22}$$

Let ExpBad_c *be the event that there are $J \subseteq \mathcal{X}_c^{\mathrm{pad}}$ with $1 \le |J| \le \lceil n_c/(\vartheta s_c)\rceil$ and sets $A_x \subseteq [s_c]$, $|A_x| \ge (1-4\eta)s_c$, such that*

$$\left|\bigcup_{x\in J}\{G_c(x,j) : j \in A_x\}\right| < (1-5\eta)s_c|J|.$$

Without conditioning on sampler or spine success, this event has probability at most

$$\nu_{\mathrm{exp},c} = \sum_{r=1}^{\min\{p_{\mathrm{sel}},\lceil n_c/(\vartheta s_c)\rceil\}} \binom{p_{\mathrm{sel}}}{r} 4^{-\eta s_c r} = \mathrm{negl}(\lambda). \tag{23}$$

Consequently, on SampleOK $\cap$ SpineOK $\cap$ $\mathsf{ExpBad}_c^{\mathrm{c}}$*, any sets of usable occurrences $A_i \subseteq [s_c]$ with $|A_i| \ge (1-4\eta)s_c$ satisfy*

$$\left|\bigcup_{i<r_c}\{g_{c,id_i}(j) : j \in A_i\}\right| \ge (1-5\eta)r_c s_c. \tag{24}$$

Proof. The exposure reduction has four deterministic steps once its random function is fixed.

1. Implement the shard oracle by sampling its entire s_c-coordinate vector when a first argument is queried for the first time, while revealing to the machine only the requested coordinates.

2. Expose every distinct first argument queried by the received experiment and its replay, all reached spine searches, and a pruned canonical checkpoint wrapper. On $\mathsf{SampleOK} \cap \mathsf{SpineOK}$, also expose the selected r_c identities.

3. Let $q_{\mathrm{pre}} = q_{\mathrm{pre}}(\lambda)$ be a deterministic polynomial upper bound on the number of distinct shard-oracle first arguments made by the polynomial-time received experiment and its initialization replay. Take p_{sel} to be a deterministic integer upper bound on

$$q_{\mathrm{pre}} + 3(K_{\mathrm{sp}} + 1)\mu C_{\mathrm{tr}} + \mu.$$

   Such a bound exists independently of the experiment outcome. Pad every outcome with the length-lexicographically first fresh dummy inputs and denote the resulting p_{sel}-element input set by $\mathcal{X}_c^{\mathrm{pad}}$.

4. Defer every shard-oracle query made only inside a restored local trial until after the preceding exposure. Then execute each omitted trial from its saved configuration.

The final reordering does not change any answer because the oracle is a fixed function; cache membership is hidden, and discarded trial configurations are not retained. Thus it preserves the joint law of the experiment, the search, the selected identities, and the raw output. By Equation (19), polynomiality of the remaining factors, and $s_c \geq \kappa$,

$$\log_2 p_{\mathrm{sel}} \leq \log_2 K_{\mathrm{sp}} + O(\log \lambda) = o(\eta\kappa) = o(\eta s_c).$$

Also $\eta s_c = \omega(\log \lambda)$. Apply Lemma 4.9 unconditionally with $(m, s, T) = (n_c, s_c, p_{\mathrm{sel}})$. This proves Equation (23); the simultaneous retained-set conclusion gives Equation (24) even though the sets A_i are determined after the rewinds. □

We now define the partial word that both recovery callers compute. For every integer coordinate $q \in [Rn_c]$, take the lexicographically first populated raw candidate (i, j) carrying coordinate q, and write its value at the evaluation point $z_{\ell_c,q}$. Write Y_c for the resulting partial word, $P_c = \mathrm{supp}(Y_c)$, and, only for analysis, let e_c be the number of supported values that disagree with $C_{n,c}$. If $d_c > 0$, let I_c be the first d_c points of $V_{\ell_c} \setminus P_c$ in public order; if $d_c = 0$, let $I_c = \varnothing$. For $v \in \mathbb{F}^{d_c}$, $\mathsf{Fill}_c(Y_c, I_c, v)$ agrees with Y_c on P_c, with v on I_c, and is $\perp$ elsewhere. Mini candidates occupy their fixed first $2n_c$ evaluation points, and tiny candidates form a partial raw message. None of these operational maps computes e_c.

Lemma 4.11 (Effective first-write support)**.** *On* $\mathsf{Good}_{\mathrm{loc}} \cap \bigcap_{c \text{ sampled}} \mathsf{ExpBad}_c^{\mathsf{c}}$*, every sampled chunk satisfies*

$$|P_c| - 2e_c \geq c_c = \min\{n_c, \lceil \vartheta\mu s_c \rceil\}. \tag{25}$$

The public complement I_c exists on every raw record, not merely on this good event. Both paired recovery callers compute identical Y_c, P_c, I_c. Furthermore every mini partial word has support size m_c' and error count e_c' satisfying $m_c' - 2e_c' \geq n_c$, and every tiny message is correct.

Proof. By definition, $\mathsf{Good}_{\mathrm{loc}}$ implies $\mathsf{SampleOK} \cap \mathsf{SpineOK}$, so Lemma 4.10 applies. Fix a sampled chunk and write $q = r_c s_c$. For each $i < r_c$, let A_i be its usable occurrences. There are at most $4\eta q$ nonusable occurrences in total. By Equation (24), the usable occurrences occupy at least $(1 - 5\eta)q$ distinct coordinates. A nonusable occurrence can precede a usable one and corrupt the selected

value at at most one coordinate, and every erroneous selected value is charged to a nonusable occurrence. Hence $e_c \le 4\eta q$ and

$$|P_c| - 2e_c \ge (1 - 5\eta)q - 8\eta q = (1 - 13\eta)q = \vartheta r_c s_c.$$

The left side is integral. If $r_c = \mu$, this gives $\lceil \vartheta \mu s_c \rceil$, capped by n_c. If $r_c < \mu$, the definition of r_c gives $\vartheta r_c s_c \ge n_c$. This proves Equation (25).

If $d_c > 0$, then $r_c = \mu$ and $\mu s_c < n_c/\vartheta < 2n_c$. Thus $|P_c| < 2n_c$. Since $R \ge 3$, at least $Rn_c - |P_c| \ge n_c \ge d_c$ public evaluation points remain, so I_c exists. If $d_c = 0$ the claim is immediate. Determinism of the shared spine, raw record, and public first-coordinate rule proves paired equality.

For a mini chunk, at least $(1 - 4\eta)2n_c$ of the fixed coordinate occurrences are usable. If w'_c positions are missing, then $e'_c + w'_c \le 4\eta(2n_c)$, whence

$$m'_c - 2e'_c = 2n_c - w'_c - 2e'_c \ge 2n_c - 2(e'_c + w'_c) \ge (1 - 8\eta)2n_c \ge n_c.$$

The tiny conclusion is part of the local extraction contract. □

The final algorithm sends only values at the public complement; in particular, it never transmits support indices.

4.6 Paired recovery and conclusion

For a public allocation, serialize the complement values in the fixed order of $\mathcal{K}(u)$, using exactly B bits per field element. The inverse parser expects precisely $B \sum_{c \text{ sampled}} d_c$ bits and returns $\perp$ on any other length. The compressor may replay $\mathcal{A}_1$ from tr to recover $(n, \varphi_n, DB^*, st_{\mathcal{A}})$; after this replay, both callers form the same extraction state $\mathfrak{X}$.

TwoForestRecovery: Paired no-index compression and error-and-erasure extraction.
Input: The compressor entry receives $(1^\lambda, S, \mu, tr; \rho)$ and is parameterized by the fixed adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$. The extractor entry receives the continuation $\mathcal{A}_2(st_{\mathcal{A}})$ and $(1^\lambda, n, S, \mu, \varphi_n, DB_{\text{short}}; \rho)$. Both have the same ambient public parameters and persistent random functions.
Output: The compressor returns DB_{short}; the extractor returns DB_{ext} or $\perp$.
Guarantee: On paired inputs, the two entries use the same raw record and public complement sets. The extractor has no transcript or original-database input.

1. The compressor replays $\mathcal{A}_1$ from tr to recover $(n, \varphi_n, DB^*, st_{\mathcal{A}})$; the extractor uses its explicitly supplied n, φ_n and continuation. Both entries then parse φ_n, compute $u = n - n_0$, $\mathcal{K}(u)$, the classes, and the allocation Equation (15). On a failed replay, malformed public data, or incorrectly sized ρ, the compressor returns the empty string and the extractor returns $\perp$. Otherwise each invokes AdaptiveRawExtract$(\mathfrak{X}, \rho)$ and constructs $(Y_c, P_c, I_c)_c$ by the public first-coordinate rule.

2. The compressor obtains $(x_{n,c})_c = \mathsf{TwoForestSplit}_{n_0,u}(DB^*)$. For each sampled c, set
$$C_{n,c} := \text{AdditiveEC}(\mathsf{encode}, (\ell_c, x_{n,c}))$$
and record $C_{n,c}[I_c]$. Return the concatenation of these vectors in public chunk order.

3. The extractor parses the advice into vectors $(v_c)_c$. For sampled c, set
$$x'_c := \text{AdditiveEC}(\mathsf{decode}, (\ell_c, \mathsf{Fill}_c(Y_c, I_c, v_c))).$$
For mini c, set $x'_c := \text{AdditiveEC}(\mathsf{decode}, (\ell_c, Y_c))$ for its partial redundancy-two word. For tiny c, set x'_c to its raw message if every coordinate is present and to $\perp$ otherwise. Return $\mathsf{TwoForestJoin}_{n_0,u}((x'_c)_c)$ if every $x'_c \neq \perp$, and return $\perp$ otherwise.

The following lemma combines the deterministic decoding condition with the probability and advice accounting established above.

Lemma 4.12 (Paired recovery format, decoding, and resources)**.** *Under all recovery hypotheses of Theorem 1.1, the two entries of* TwoForestRecovery *are total. On paired inputs, the advice length is exactly*
$$B \sum_{c \text{ sampled}} d_c \leq B \max\{n - (1-\epsilon)\mu S, 0\}, \tag{26}$$
and the advice contains no indices. The extractor runs in time quasi-polynomial in $(\lambda, t_{\mathcal{A}})$ and satisfies
$$\Pr[\mathsf{Pass} \wedge (DB_{\text{ext}} \neq DB^*)] \leq \text{negl}(\lambda). \tag{27}$$

Proof. All parsers, searches, rewinds, completion maps, decoders, and malformed-input branches have finite public bounds, so both entries are total. Fix the good events in Lemma 4.11. For a sampled chunk, paired advice supplies the correct codeword values on I_c, disjoint from P_c. The filled word has $|P_c| + d_c$ supported entries and exactly e_c errors. By Equation (25),
$$|P_c| + d_c - 2e_c \geq c_c + (n_c - c_c) = n_c.$$

The decoder guarantee in Lemma 4.4 therefore returns $x_{n,c}$. The mini inequality and exact tiny opening in Lemma 4.11 give the same conclusion for those chunks. Lemma 4.3 then makes the final join equal DB^*.

For the advice count, Equation (8) gives

$$c_c \geq n_c \min\left\{1, \frac{\vartheta \mu S}{n}\right\}$$

on every sampled chunk, while mini and tiny chunks use no advice. Since the chunk lengths sum to n,

$$\sum_{c \text{ sampled}} d_c \leq \max\{n - \vartheta\mu S, 0\} \leq \max\{n - (1-\bar{\epsilon})\mu S, 0\} \leq \max\{n - (1-\epsilon)\mu S, 0\},$$

where $\vartheta = 1 - 13\bar{\epsilon}/40 \geq 1 - \bar{\epsilon}$ and $\bar{\epsilon} \leq \epsilon$. This proves Equation (26), including zero advice in the saturated regime.

There are at most $2(1 + \lfloor \log_2 N \rfloor)$ chunks. Combining Definition 4.7 and lemmas 4.6, 4.8 and 4.10 bounds the bad event jointly with Pass by

$$2^{-\lambda-1} + \frac{1}{K_{\text{sp}} + 1} + K_{\text{sp}} q_{\text{loc}} \left(\frac{8N}{T_{\text{ext}} + 1} + (1-\eta)^{\kappa}\right) + \frac{K_{\text{sp}}\mu}{T_{\text{ext}} + 1} + \nu_{\text{bind}}(\lambda) + \sum_{c \text{ sampled}} \nu_{\text{exp},c}(\lambda). \quad (28)$$

Every term is negligible by Equation (19), Equation (23), polynomiality of N, μ, and quasi-polynomial binding. Outside this event the deterministic decoding above succeeds, proving Equation (27).

The accepting-spine and rewind work is quasi-polynomial by Lemma 4.8. Completion scans $O(Rn_c)$ coordinates per chunk, and Lemma 4.4 bounds decoding by

$$B^{O(1)} \sum_{c \in \mathcal{K}(u)} (Rn_c)^4$$

bit operations. The field-size, public-size, and polynomial-redundancy conditions make this polynomial in the displayed parameters; moreover $B \leq t_{\mathcal{A}}$ because the experiment writes a nonempty B-bit-block database. Hence the total extractor time is quasi-polynomial in $(\lambda, t_{\mathcal{A}})$. Deterministic paired equality from Lemmas 4.8 and 4.11 shows that no support metadata or transcript is needed by the extractor. □

We can now substitute the rounded public slack and finish the theorem.

Proof of Theorem 1.1. The definition of $\bar{\epsilon}$ gives $\epsilon/2 < \bar{\epsilon} \leq \epsilon$. Consequently

$$R = \Theta(1/\bar{\epsilon}) = \Theta(1/\epsilon), \qquad \zeta_0 = 2^{-(k_{\text{rec}}+3)} \in (\epsilon/16, \epsilon/8].$$

The dyadic parameter has a canonical encoding of $O(1 + \log(1/\epsilon))$ bits. Since $R\widehat{N} \leq 2^B$, one has $B = \Omega(\log N + \log(1/\epsilon))$, so $z_0 = O(B)$. Substitution into Equation (14) proves $O(BU \log N/\epsilon)$ simplified publisher work for every U, and the bit work, storage, and tree-depth conclusions follow from Lemma 4.5. The proof counts only suffix carries and therefore introduces no relation between n_0 and U. The same lemma proves honest correctness and, under the additional recovery block-size condition, the node-state and externally-materialized output bounds.

For recovery, the extra hypothesis implies $\bar{\epsilon}\kappa = \omega(\log \lambda)$. Also $R \geq 12\mathrm{e}/\eta$, $\vartheta > 1/2$, and all recovery-validity and efficiency conditions needed by Lemma 4.12 hold uniformly at every legal length. That lemma gives the exact paired interfaces, advice bound, quasi-polynomial extractor, and negligible joint error asserted in the theorem. Its construction and proof use pairwise distinct identities and make no replication claim, which establishes the stated scope. □

References

[ABC+07] Giuseppe Ateniese, Randal C. Burns, Reza Curtmola, Joseph Herring, Lea Kissner, Zachary N. J. Peterson, and Dawn Xiaodong Song. Provable data possession at untrusted stores. In *Proceedings of the 14th ACM Conference on Computer and Communications Security*, pages 598–609. ACM, 2007.

[BS80] Jon Louis Bentley and James B. Saxe. Decomposable searching problems I: Static-to-dynamic transformation. *Journal of Algorithms*, 1(4):301–358, 1980.

[CKW13] David Cash, Alptekin Küpçü, and Daniel Wichs. Dynamic proofs of retrievability via oblivious RAM. In *Advances in Cryptology – EUROCRYPT 2013*, volume 7881 of *Lecture Notes in Computer Science*, pages 279–295. Springer, 2013.

[JJ07] Ari Juels and Burton S. Kaliski Jr. PORs: Proofs of retrievability for large files. In *Proceedings of the 14th ACM Conference on Computer and Communications Security*, pages 584–597. ACM, 2007.

[KCR19] Swanand Kadhe, Jichan Chung, and Kannan Ramchandran. SeF: A secure fountain architecture for slashing storage costs in blockchains. arXiv preprint arXiv:1906.12140, 2019.

[MJS+14] Andrew Miller, Ari Juels, Elaine Shi, Bryan Parno, and Jonathan Katz. Permacoin: Repurposing Bitcoin work for data preservation. In *2014 IEEE Symposium on Security and Privacy*, pages 475–490. IEEE, 2014.

[PLBD18] Doriane Pérard, Jérôme Lacan, Yann Bachy, and Jonathan Detchart. Erasure code-based low storage blockchain node. In *2018 IEEE International Conference on Internet of Things (iThings), Green Computing and Communications (GreenCom), Cyber, Physical and Social Computing (CPSCom), and Smart Data (SmartData)*, pages 1622–1627. IEEE, 2018.

[Rab89] Michael O. Rabin. Efficient dispersal of information for security, load balancing, and fault tolerance. *Journal of the ACM*, 36(2):335–348, 1989.

[RS60] I. S. Reed and G. Solomon. Polynomial codes over certain finite fields. *Journal of the Society for Industrial and Applied Mathematics*, 8(2):300–304, 1960.

[RV21] Ravi Kiran Raman and Lav R. Varshney. Coding for scalable blockchains via dynamic distributed storage. *IEEE/ACM Transactions on Networking*, 29(6):2588–2601, 2021.

[SSM26] Elaine Shi, Rose Silver, and Changrui Mu. Decentralized data archival: New definitions and constructions. In Shubhangi Saraf, editor, *17th Innovations in Theoretical Computer Science Conference (ITCS 2026)*, volume 362 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 116:1–116:22, Dagstuhl, Germany, 2026. Schloss Dagstuhl

– Leibniz-Zentrum für Informatik. Full version: Cryptology ePrint Archive, Paper 2025/969.

[SSP13] Elaine Shi, Emil Stefanov, and Charalampos Papamanthou. Practical dynamic proofs of retrievability. In *Proceedings of the 2013 ACM SIGSAC Conference on Computer and Communications Security*, pages 325–336. ACM, 2013.