Benchmarking Blockchain Data Availability: A Comparative Empirical Study of Six Production DA Protocols

Prasad Kumkar, Pranjal Sarode
Chainscore Labs, Pune, IN
{prasad, pranjal}@chainscore.finance

***Abstract*—Data availability (DA) is a critical bottleneck in blockchain scalability, yet no empirical cross-protocol comparison exists using production network data. This paper presents the first comprehensive benchmarking study of six DA protocols (Polkadot ELVES, Ethereum EIP-4844/PeerDAS, Celestia, Espresso Tiramisu, NEAR, and Avail) using 90 days of mainnet data (November 2025–February 2026). We introduce a taxonomy that classifies DA architectures along three paradigms: Data Availability Sampling (DAS), Verifiable Information Dispersal (VID), and validator-set recovery. Our empirical analysis measures throughput (MiB/s), cost ($/MiB), utilization, and block dynamics across all six protocols under a unified methodology. Key findings include: (i) all protocols operate well below saturation (<50% utilization), making current measurements demand-constrained rather than supply-constrained; (ii) DA costs span seven orders of magnitude across protocols, from nano-dollars (Espresso) to cents per MiB (Celestia); (iii) no protocol implements slashing for pure data withholding, with enforcement remaining indirect across all designs; and (iv) DA is universally treated as temporary, with retention windows ranging from 85 minutes to 18 days. We release all collection tooling, datasets, and analysis notebooks under open licenses to enable reproducibility.**

***Index Terms*—data availability, blockchain scalability, erasure coding, data availability sampling, benchmarking, rollups**

I. Introduction

Blockchain scalability hinges on a deceptively simple requirement: every participant must be confident that block data has been published and is retrievable. This property, known as *data availability* (DA), underpins the security of rollups, parachains, and sharded architectures alike. Without DA guarantees, validators cannot verify state transitions, light clients cannot detect fraud, and the trust model of the entire system collapses to that of a permissioned committee [1].

The past three years have witnessed a proliferation of DA solutions. Ethereum introduced blob-carrying transactions (EIP-4844) and is progressively deploying PeerDAS for cell-level sampling [2], [3]. Celestia launched the first dedicated DA network with two-dimensional erasure coding and namespace-filtered sampling [4]. Polkadot's ELVES protocol distributes erasure-coded parachain data across its validator set with approval-gated finality [5]. Espresso's Tiramisu introduces a three-layer Verifiable Information Dispersal (VID) architecture that explicitly rejects DAS in favor of algebraic dispersal proofs [6]. NEAR provides sharding-based DA through Nightshade with a rollup-oriented blob store [7], and Avail combines KZG polynomial commitments with DAS-enabled light clients on a Substrate-based chain [8].

Despite this activity, the field lacks an empirical, cross-protocol comparison grounded in production data. Existing comparisons are either qualitative [9], vendor-authored, or limited to theoretical analysis of individual protocols. No study has simultaneously measured throughput, cost, and utilization across multiple DA networks under a unified methodology. This gap is consequential: rollup developers choosing a DA backend, protocol designers evaluating trade-offs, and researchers assessing the state of the art all lack a common empirical baseline.

This paper fills that gap. Our contributions are:

1) **A taxonomy of DA architectures** that classifies protocols along three paradigms (Data Availability Sampling, Verifiable Information Dispersal, and validator-set recovery) and along orthogonal dimensions of encoding scheme and commitment type (Section II).
2) **A qualitative architecture comparison** of encoding redundancy, commitment schemes (KZG vs. NMT vs. Merkle), and verification mechanisms across all six protocols, grounded in protocol specifications and documentation (Section III).
3) **An empirical benchmarking study** using 90 days of production mainnet data (November 2025–February 2026) across Polkadot, Ethereum, Celestia, Espresso, NEAR, and Avail, measuring throughput (MiB/s), cost ($/MiB), utilization (%), and block dynamics under consistent definitions (Section V).
4) **A comparative security analysis** of trust assumptions, Byzantine fault tolerance thresholds, data withholding enforcement, and retention policies across all six protocols (Section VI).
5) **Open-source tooling and datasets**: all data collection scripts, raw datasets, and analysis notebooks are released under Apache 2.0 (code) and CC BY 4.0 (data) licenses to enable reproducibility and longitudinal extension.

The remainder of this paper is organized as follows. Section II provides background on the DA problem, introduces our taxonomy, describes each protocol, and positions our work relative to prior literature. Section III presents a qualitative comparison of protocol architectures. Section IV details our data collection and analysis methodology. Section V presents

empirical results organized by metric. Section VI analyzes security properties and trust assumptions. Section VII discusses trade-offs, implications, and limitations. Section VIII concludes with a summary of findings and directions for future work.

II. Background and Related Work

A. Data Availability Fundamentals

Data availability (DA) must be distinguished from the related but distinct problems of data *storage* and data *retrieval*. Storage concerns the persistence of data over time; retrieval concerns the ability to read stored data on demand. DA addresses a different question: *was the data published at all?* Formally, a block producer who withholds part of a block's data can cause validators and light clients to accept an invalid state transition, because the withheld data may contain a fraudulent transaction whose invalidity can never be proven if the data is unavailable [10].

Definition 1 (Data Availability)**.** *A block B satisfies the data availability property if every honest participant can retrieve the complete block data within a bounded time window Δ after the block header is published.*

A DA scheme must provide four properties:

1) **Completeness.** All data constituting block B is published to the network. A DA scheme is complete if no honest node accepts a block header unless the corresponding data has been made available to the network.
2) **Soundness.** The encoding of B is verifiably correct. Given a commitment $C(B)$ included in the block header, any participant can verify that a received coded piece is consistent with $C(B)$ without possessing the entire block.
3) **Availability detection.** Resource-constrained participants (light clients) can assess whether the data behind a block header is available without downloading the full block. This property separates DA from simple data retrieval.
4) **Reconstruction.** The original data B can be recovered from any sufficiently large subset of coded pieces. If $E(B)$ denotes the erasure-coded encoding of B producing n coded pieces with redundancy factor r, then B is reconstructible from any n/r pieces.

Notation. Throughout this paper we use the following conventions. A block's raw data is denoted B, with $|B|$ its size in bytes. The encoding function $E(B)$ produces n coded pieces (variously called shares, chunks, or cells depending on the protocol). A commitment $C(B)$ is included in the block header and serves as a binding digest of $E(B)$. Verification $V(C, s_i)$ checks that coded piece s_i is consistent with commitment C. The redundancy factor $r = n \cdot |s_i|/|B|$ captures the total storage expansion introduced by the encoding.

B. Taxonomy of DA Approaches

We classify the six protocols along three orthogonal dimensions: the *availability verification paradigm* (how clients gain confidence that data is available), the *encoding scheme* (how raw data is expanded into coded pieces), and the *commitment scheme* (how coded pieces are bound to the block header).

1) Availability Verification Paradigm: Three distinct paradigms exist.

a) DAS-based protocols (Avail, Celestia, Ethereum/PeerDAS): In Data Availability Sampling (DAS), light clients randomly request a small number of coded pieces from the network. If all requested pieces are returned with valid proofs, the client concludes with high statistical confidence that the full data is available. The security argument relies on the *honest minority assumption*, requiring at least one honest, non-eclipsed full node to be reachable. This ensures that correctly encoded pieces are seeded into the peer-to-peer network. Confidence grows exponentially with sample count. Celestia's light clients achieve approximately 99% confidence with 16 samples from a two-dimensional encoding [4], while Avail targets 99.9% confidence with 10 samples from a one-dimensional encoding [8].

b) Validator-set recovery protocols (Polkadot, NEAR): The full validator set collectively stores erasure-coded chunks, and availability is attested through consensus-level mechanisms rather than client-side sampling. No light client sampling protocol exists; instead, confidence in data availability is anchored in the consensus and finality guarantees of the network. Polkadot requires a two-thirds supermajority of validators to sign availability bitfield votes before a parachain block is considered available, with availability gating the approval checking and finality pipeline [5]. NEAR distributes erasure-coded chunks to validators within each shard, relying on per-shard chunk producer accountability within the Nightshade sharding protocol [7].

c) VID-based protocols (Espresso/Tiramisu): Verifiable Information Dispersal (VID) combines erasure coding with algebraic proofs that each recipient received a valid piece. Espresso's Tiramisu architecture uses KZG polynomial commitments paired with vector commitments to produce a DA certificate: a threshold signature from a DA committee attesting that all coded pieces were correctly distributed [6]. The Espresso design explicitly rejects DAS, arguing that client-side sampling introduces "unnecessary redundancies" when a committee can collectively certify dispersal. Light client confidence in Tiramisu derives from finality combined with the DA certificate, rather than from probabilistic sampling.

2) Encoding Scheme: The choice of erasure coding determines the redundancy factor, reconstruction threshold, and proof structure.

a) One-dimensional Reed–Solomon (1D RS): Avail, Polkadot, Ethereum (PeerDAS), and NEAR all employ 1D Reed–Solomon encoding. The block data is split into k pieces and extended to $n = r \cdot k$ coded pieces, where r is the redundancy factor. Any k of the n pieces suffice for reconstruction. Redundancy factors vary: Ethereum uses $r = 2$ (128 columns, 64 original) [3], Avail uses approximately $r = 2$ (row-wise extension) [8], and Polkadot and NEAR use approximately $r = 3$ [5], [7].

b) Two-dimensional Reed–Solomon (2D RS): Celestia arranges shares into a $k \times k$ square and extends in both

dimensions to produce a $2k \times 2k$ extended data square, yielding a $4\times$ storage expansion [4]. The 2D structure enables row-wise and column-wise sampling, and any full row or column can be independently reconstructed and verified. This design provides stronger fraud proof properties (an invalid encoding can be proven with a single row or column) at the cost of higher total redundancy relative to 1D schemes.

c) Polynomial VID: Espresso treats the encoding as evaluations of a polynomial over a finite field. Each coded piece is a point evaluation, and decoding is polynomial interpolation. The KZG commitment to the polynomial serves simultaneously as the encoding commitment and the basis for per-piece verification proofs [6].

3) Commitment Scheme: The commitment scheme determines proof sizes, verification costs, and trust assumptions.

a) KZG polynomial commitments (Avail, Ethereum, Espresso): Kate–Zaverucha–Goldberg (KZG) commitments [11] produce constant-size commitments and constant-size opening proofs regardless of the polynomial degree. Verification requires a single bilinear pairing check. The principal drawback is the requirement for a trusted setup ceremony to generate structured reference strings; both Ethereum and Avail have conducted such ceremonies.

b) Namespaced Merkle Trees (Celestia): Celestia uses Namespaced Merkle Trees (NMTs), which augment standard Merkle trees with namespace identifiers at each leaf [4]. NMTs enable namespace-filtered retrieval: a light client interested in a specific application namespace can request and verify only the shares belonging to that namespace, with a proof of completeness. Proofs are logarithmic in the number of shares. No trusted setup is required.

c) Plain Merkle commitments (Polkadot, NEAR): Polkadot and NEAR use standard Merkle trees to commit to erasure-coded chunks [5], [7]. Proofs are logarithmic in the number of chunks. No trusted setup is required, and the scheme is simple to implement and audit, but it lacks the constant-size proof property of KZG and the namespace-filtering capability of NMTs.

C. Protocol Descriptions

We provide concise factual descriptions of each protocol's DA mechanism as deployed during our measurement window (November 2025–February 2026).

a) Polkadot (ELVES): Polkadot's relay chain distributes erasure-coded Proof-of-Validity (PoV) data across approximately 600 validators using 1D Reed–Solomon encoding with roughly $3\times$ redundancy and Merkle tree commitments [5], [12]. A parachain block is deemed available when at least two-thirds of validators sign availability bitfield votes. The block then enters the approval checking and GRANDPA finality pipeline. The relay chain supports up to 100 cores with a maximum PoV size of 10 MiB per core, producing relay blocks approximately every 6 seconds. DA is priced through the coretime allocation mechanism rather than per-byte fees. Erasure-coded chunks are retained for approximately 25 hours.

b) Ethereum (EIP-4844 + PeerDAS): Ethereum's DA layer uses blob-carrying transactions introduced by EIP-4844 [2], with blobs committed via KZG polynomial commitments. A separate blob gas market, governed by an EIP-1559-style exponential feedback mechanism, prices blob inclusion independently of execution gas. As of the Pectra upgrade (EIP-7691 [13]), the network targets 6 blobs per block with a maximum of 9, each blob being 128 KiB. PeerDAS (EIP-7594 [3]) introduces cell-level sampling with 1D Reed–Solomon encoding at $2\times$ redundancy over 128 columns, with a custody requirement of 4 columns per node. Blocks are produced every 12 seconds, with Casper FFG finality requiring approximately 12.8 minutes (two epochs). Blobs are retained for approximately 18 days (4,096 epochs) before consensus-layer pruning.

c) Celestia: Celestia constructs a two-dimensional Reed–Solomon extended data square: a $k \times k$ original data square is extended to $2k \times 2k$, yielding a $4\times$ share expansion [4]. Shares are committed using Namespaced Merkle Trees, enabling namespace-filtered retrieval for application-specific data. Light clients perform DAS by randomly sampling shares and verifying NMT inclusion proofs, achieving statistical confidence in data availability without downloading the full block. Celestia runs CometBFT (Tendermint) consensus with single-slot finality, producing blocks every 6 seconds. The current maximum block size is 8 MiB (a 128×128 original data square), with a 30-day data availability window.

d) Espresso (Tiramisu): Espresso's Tiramisu architecture comprises three layers. Savoiardi provides a VID layer with bribery-resistant dispersal, Mascarpone operates a DA committee for fast retrieval, and Cocoa serves as an untrusted CDN for bulk delivery [6]. The VID layer uses KZG polynomial commitments combined with vector commitments to produce per-piece validity proofs. HotShot consensus provides responsive finality (no fixed block time), achieving approximately 2-second finality with 100 nodes. Mainnet 0 operates with 20 operators running 100 nodes, with no slashing mechanism currently active. Data retention ranges from 1 to 7 days depending on configuration.

e) NEAR: NEAR's Nightshade sharding protocol distributes erasure-coded chunk data across validators, with each shard's chunk producers responsible for encoding and distributing their shard's data [7]. For rollup use cases, NEAR provides a DA interface through a blob store smart contract. Data is submitted as function call input and subsequently pruned from state after the garbage collection window of approximately 2.5 days (5 epochs). NEAR uses 1D Reed–Solomon encoding with approximately $3\times$ redundancy and Merkle commitments. Blocks are produced approximately every second, with a maximum upload of 4 MB per function call. NEAR does not implement DAS or an active data withholding challenge system; availability confidence derives from the validator set's consensus participation.

f) Avail: Avail combines KZG polynomial commitments with 1D Reed–Solomon encoding (row-wise, approximately $2\times$ redundancy) to enable full DAS [8], [14]. Light clients participate in a DHT-based sampling network, requesting random cells and verifying KZG opening proofs to gain confidence in

data availability. The chain runs BABE block production and GRANDPA finality (Substrate-based), producing blocks every 20 seconds with a maximum block size of up to 64 MiB. Application-Specific Data Retrieval (ASDR) allows clients to filter and download only the data relevant to their application identifier. The default data retention window is 256 blocks (approximately 85 minutes), with the VectorX bridge to Ethereum providing DA attestations with an approximate 2-hour delay.

D. Related Work

a) Foundational theory: Al-Bassam, Sonnino, and Buterin [10] introduced the concept of fraud and data availability proofs, establishing the theoretical foundation for DAS. By erasure-coding block data and enabling light clients to randomly sample coded pieces, a network can guarantee data availability with high probability under an honest minority assumption. This work underpins the DAS-based protocols (Celestia, Avail, and Ethereum's PeerDAS) studied in this paper. The KZG polynomial commitment scheme [11], originally proposed by Kate, Zaverucha, and Goldberg, provides the constant-size commitments and opening proofs used by Avail, Ethereum, and Espresso for efficient per-piece verification.

b) Protocol-specific designs: Al-Bassam's LazyLedger proposal [4] introduced the concept of a dedicated DA blockchain with namespace-filtered retrieval, which evolved into Celestia. Burdges and Cevallos [5] formalized Polkadot's ELVES protocol for availability and validity, proving security under Byzantine assumptions with approval-based checking. Espresso Systems [6] describe the HotShot consensus and Tiramisu DA architecture, with further analysis of the VID construction and its bribery-resistance properties presented at ITCS 2026 [15]. Ethereum has progressed from monolithic blob transactions (EIP-4844 [2]) through blob throughput increases (EIP-7691 [13]) to full peer-level DAS (EIP-7594 [3]), following an incremental deployment path toward Dankssharding [16].

c) Surveys and comparisons: Saif et al. [1] provide a survey of data availability mechanisms in the context of Layer 2 rollups, classifying approaches by their trust assumptions and encoding strategies. Their analysis is qualitative and does not include empirical measurements from production networks. Symbolic Capital [9] published a qualitative comparison of DA solutions covering design trade-offs, but likewise without empirical data on throughput, cost, or utilization. Several vendor-authored blog posts and documentation pages [14], [17] describe individual protocols but do not offer cross-protocol comparisons under a unified methodology.

d) Positioning of this work: This paper is the first to combine empirical production data across six DA protocols simultaneously under a consistent measurement framework. Prior work has focused on a single protocol, offered qualitative taxonomies without empirical grounding, or relied on testnet or simulation data rather than production mainnet measurements. Our study complements the theoretical foundations laid by [10] and the qualitative surveys of [1], [9] by providing the empirical baseline that these works identify as missing from the literature.

III. Protocol Architecture Comparison

This section presents a qualitative comparison of the six DA protocols along three architectural dimensions: encoding and redundancy schemes, commitment and proof systems, and verification paradigms. These dimensions collectively determine each protocol's trade-off surface between bandwidth efficiency, light client capability, trust assumptions, and fault tolerance.

A. Encoding and Redundancy

All six protocols employ Reed–Solomon (RS) erasure coding to introduce redundancy into block data, but they differ substantially in dimensionality, redundancy factor, and reconstruction semantics. Table I summarizes these differences.

The central trade-off is between redundancy overhead (bandwidth and storage cost) and reconstruction robustness. Higher redundancy factors require proportionally more network bandwidth for data distribution and more aggregate storage across the validator or node set, but they provide stronger guarantees that data can be recovered even when a larger fraction of coded pieces is unavailable.

Celestia exhibits the highest redundancy at $\sim 4\times$ due to its two-dimensional RS structure: original data of dimension $k \times k$ is extended to $2k \times 2k$, yielding four times the original data volume in shares [4]. This overhead is the direct cost of enabling two-dimensional random sampling, which allows light clients to verify availability by sampling from both rows and columns of the extended matrix. The 2D structure also permits row- or column-level reconstruction: any k shares from a single row (or column) of length $2k$ suffice to recover that row (or column), enabling localized repair without reconstructing the entire block.

Polkadot and **NEAR** both employ $\sim 3\times$ redundancy, distributing erasure-coded pieces across their respective validator sets. In Polkadot's ELVES protocol, parachain block data (Proofs of Validity, or PoVs) are RS-encoded and distributed such that each validator stores one coded chunk, with reconstruction requiring $f+1$ of n chunks where $n = 3f + k$ [5], [12]. NEAR's Nightshade sharding similarly encodes chunk data with $\sim 3\times$ redundancy, requiring only $1/3$ of the parts for reconstruction [7]. In both cases, the redundancy distributes data responsibility across the full validator set rather than relying on a smaller committee or random sampling.

Ethereum and **Avail** achieve the most bandwidth-efficient encoding at $\sim 2\times$ redundancy. Ethereum's EIP-4844 blob transactions encode data into 128-byte cells arranged across 128 columns. The emerging PeerDAS specification extends each blob's polynomial evaluation points such that any 64 of the 128 columns suffice for reconstruction [2]. Avail performs row-wise RS encoding on its application data matrix, extending n original pieces to $2n$ coded pieces per row, so that any n of the $2n$ pieces suffice for recovery [8]. The $2\times$ factor represents the minimum redundancy that still provides meaningful erasure resilience, and is sufficient for DAS-based verification when combined with appropriate commitment schemes.

TABLE I: Encoding and Redundancy Comparison Across DA Protocols

Protocol	Encoding Scheme	Redundancy Factor	Reconstruction Threshold
Polkadot [5]	1D RS across validator set	$\sim 3\times$	$f+1$ of n validators ($n = 3f + k$)
Ethereum [2]	1D RS (PeerDAS cells)	$\sim 2\times$	50%+ of 128 columns
Celestia [4]	2D RS ($k \times k \to 2k \times 2k$)	$\sim 4\times$ (in shares)	$k \times k$ unique shares
Espresso [6]	Polynomial VID (eval/interp)	$\sim 1/r$ (configurable, $r = m/n$)	m of n evaluations
NEAR [7]	1D RS (chunk parts)	$\sim 3\times$ (current)	$1/3$ of parts
Avail [8]	1D RS (row-wise)	$\sim 2\times$	n of $2n$ coded pieces

Espresso's Tiramisu uses a polynomial-based Verifiable Information Dispersal (VID) scheme rather than a traditional RS code operating on byte-level shares. Data is represented as a polynomial that is evaluated at n distinct points (one per committee member); any m evaluations suffice for interpolation, where the code rate $r = m/n$ is configurable [6]. This approach is algebraically equivalent to RS coding but is tightly integrated with the VID commitment layer, enabling a unified proof of correct dispersal that simultaneously certifies both encoding correctness and data distribution.

B. Commitment Schemes and Proof Systems

The choice of cryptographic commitment scheme determines what properties light clients can verify, what trust assumptions are required, and the computational cost of proof generation and verification. The six protocols employ three fundamentally different commitment families: KZG polynomial commitments, Namespaced Merkle Trees (NMTs), and standard Merkle trees. Table II compares their properties.

KZG polynomial commitments [11] are used by Avail, Ethereum, and Espresso. A KZG commitment to a polynomial $p(x)$ of degree d is a single elliptic curve group element ($\sim$48 bytes on BLS12-381), and an opening proof for any evaluation $p(z) = v$ is also a single group element. Verification requires two pairing operations, which are computationally more expensive than hash evaluations but yield $O(1)$-size proofs regardless of the data size. KZG commitments enable *algebraic verification of encoding correctness*: given commitments to the rows (or columns) of an erasure-coded data matrix, anyone can verify that the extended portions are consistent with the original data by checking polynomial relationships between commitments, without accessing the underlying data. This property is what enables Avail and Ethereum light clients to reject incorrectly encoded data without fraud proofs. The principal drawback is the requirement for a structured reference string (SRS) generated through a trusted setup ceremony. Compromise of the SRS toxic waste would allow an adversary to forge proofs. In practice, both Ethereum and Avail have conducted large-scale multi-party ceremonies (Ethereum's ceremony involved over 140,000 contributors) to mitigate this risk.

Namespaced Merkle Trees (NMTs) are Celestia's defining commitment innovation [4]. An NMT is a standard Merkle tree augmented with namespace identifiers: each leaf is tagged with a namespace, internal nodes carry the minimum and maximum namespace of their subtrees, and the tree is sorted by namespace. This structure enables *namespace inclusion and exclusion proofs*. A light client can request all shares belonging to a specific namespace (e.g., a particular rollup's data) and receive a Merkle proof that the returned set is complete (no shares from that namespace have been omitted, and no extraneous shares have been included). This capability is critical for Celestia's rollup-centric design, as it allows rollup nodes to efficiently retrieve only their own data without downloading the entire block. NMT proofs are $O(\log n)$ in size. Encoding correctness, however, is *not* algebraically verifiable from the NMT alone. Instead, Celestia relies on Bad Encoding Fraud Proofs (BEFPs) to detect and propagate evidence of incorrect erasure coding [10].

Standard Merkle trees are used by Polkadot and NEAR. Polkadot commits to erasure-coded PoV chunks via a Merkle trie whose root is included in the relay chain block header [12], [17]. NEAR similarly uses Merkle roots to commit to chunk data within shard blocks. Standard Merkle proofs provide $O(\log n)$ inclusion proofs and require only hash operations for verification, making them the simplest and most widely understood commitment scheme. However, plain Merkle trees provide no mechanism for light clients to verify encoding correctness or to perform data availability sampling: a Merkle proof confirms that a specific piece was included in the committed data set, but it reveals nothing about whether the overall erasure coding was performed correctly. In both Polkadot and NEAR, encoding correctness is enforced through the validator set (validators who reconstruct and re-encode can detect errors), but this verification is not accessible to light clients or external observers.

C. Verification Paradigms: DAS, VID, and Validator Recovery

The most consequential architectural distinction among DA protocols is how availability is verified. We identify three paradigms, each with different trust assumptions and light client capabilities.

1) Data Availability Sampling (DAS): DAS, as formalized by Al-Bassam et al. [10], enables light clients to verify data availability without downloading the full block. A light client randomly requests s coded pieces from the network; if all s are returned, the client gains probabilistic confidence that the full data is available. Against a $(1-q)$-withholding adversary (one that has made fraction q of the data unavailable), a client sampling s independent pieces achieves confidence $1 - q^s$.

Celestia was the first production network to deploy DAS. Its 2D RS structure means that light clients sample from the

TABLE II: Commitment Scheme Properties Across DA Protocols

Property	KZG (Avail, Ethereum, Espresso)	NMT (Celestia)	Merkle (Polkadot, NEAR)
Proof size	$O(1)$, $\sim$48 bytes	$O(\log n)$	$O(\log n)$
Trusted setup required	Yes (ceremony / SRS)	No	No
Namespace filtering	No (application-level)	Native	No
Verification cost	Pairing operations	Hash operations	Hash operations
Encoding correctness	Algebraically verifiable	Fraud proofs (BEFPs)	Not independently verifiable

$2k \times 2k$ extended square. With 16 random samples, a Celestia light client achieves approximately $1 - (1/2)^{16} \approx 99.994\%$ confidence that the data is available, assuming that at least one honest full node is non-eclipsed and participates in the peer-to-peer network [4]. The 2D structure further enables row- or column-level reconstruction: if sampling reveals a missing row, any node holding k of the $2k$ shares in that row can reconstruct it, providing a repair mechanism that does not require full block reconstruction.

Avail implements DAS over its 1D RS-encoded rows with KZG commitments. Avail light clients sample random cells from the data matrix; each cell is accompanied by a KZG opening proof. With 10 random samples, an Avail light client achieves $1-(1/2)^{10} \approx 99.9\%$ confidence $(1-1/1024)$ [8]. The use of KZG commitments provides an additional guarantee beyond sampling: the algebraic properties of polynomial commitments allow the light client to verify that each sampled cell is consistent with the committed row polynomial, effectively verifying encoding correctness without fraud proofs. Notably, Avail light clients retrieve sampled cells from a dedicated Kademlia DHT in a light-client-only P2P network before falling back to full-node RPC; cells fetched via RPC are uploaded back into the DHT, creating a self-healing replication layer that strengthens availability as light client participation grows.

Ethereum is progressively deploying DAS through the PeerDAS specification (EIP-7594). Under PeerDAS, each blob is extended from 64 to 128 evaluation points (columns), and each validator is responsible for custody of a subset of columns determined by its node ID. The current specification sets $\texttt{SAMPLES\_PER\_SLOT} = 8$ columns as the minimum custody requirement per validator [2]. Confidence in Ethereum's DAS is emergent at the system level rather than per-client: individual validators do not perform independent random sampling in the Celestia/Avail sense; instead, the aggregate custody across the validator set ensures that all 128 columns are held by at least one honest validator with high probability. This design reflects Ethereum's philosophy of leveraging its large validator set ($\sim$1,000,000 validators) rather than requiring each light client to independently sample. PeerDAS also introduces a "supernode" role: nodes that subscribe to all 128 data column subnets and can reconstruct the full data matrix from partial columns. The Fulu networking specification requires at least one supernode on the network for reconstruction of missing data, adding a qualitative trust assumption not present in the pre-PeerDAS regime where every consensus node downloaded all blob data [3].

The security of DAS across all three protocols rests on two critical assumptions: (i) at least one honest, non-eclipsed full node must hold the complete data and respond to sampling requests, and (ii) sufficient sampling participation must exist in the network to ensure that withholding is detected before the data propagation window closes. An adversary capable of eclipsing a light client's peer connections can cause the client to falsely conclude that data is available (or unavailable) regardless of the true state.

2) Validator-Set Recovery: Polkadot and NEAR adopt a different approach: the full validator set collectively stores and serves erasure-coded data, and availability is attested through consensus-integrated mechanisms rather than probabilistic sampling.

Polkadot's ELVES protocol tightly couples DA verification with its finality mechanism [5]. After a parachain collator produces a block candidate, the relay chain validators erasure-code the Proof of Validity (PoV) and distribute one chunk per validator. Each validator that successfully receives and stores its chunk signs an *availability bitfield* (a bitmap indicating which PoV chunks it holds). A PoV is considered available when $\geq 2/3$ of validators have signed its bitfield. Polkadot *gates finality on availability evidence*: the GRANDPA finality gadget will not finalize a relay chain block unless the availability bitfields for all included parachain candidates have reached the $2/3$ threshold [12], [17]. This design provides a deterministic (not probabilistic) availability guarantee conditioned on the $2/3$ honest-majority assumption, and ensures that unavailable data cannot be finalized.

NEAR's Nightshade sharding distributes chunk data across shard validators (chunk producers). Each chunk producer is responsible for producing and serving the chunk parts for its assigned shard. NEAR's availability enforcement relies on a "lazy signer" mechanism with probabilistic accountability: validators are required to attest to chunk availability, but enforcement is primarily through the consensus protocol's liveness requirements rather than explicit sampling [7]. A challenge mechanism exists in the block structure for disputing invalid or unavailable chunks, but as of our observation period, this mechanism is "not used today" in practice.

The key advantage of validator-set recovery is its tight coupling with consensus: availability evidence is a prerequisite for block finality (Polkadot) or block production (NEAR), creating a direct economic incentive for validators to store and serve data. However, *light clients cannot independently verify data availability*. A Polkadot or NEAR light client must trust that the $\geq 2/3$ validator supermajority has honestly attested to availability. There is no mechanism analogous to DAS that would allow a resource-constrained client to independently

check. Trust rests entirely on the validator set's honesty and the economic security provided by staking.

3) Verifiable Information Dispersal (VID): Espresso's Tiramisu architecture introduces a third paradigm that explicitly rejects DAS in favor of algebraic dispersal proofs [6]. Tiramisu consists of three layers:

- **Savoiardi (VID layer):** Data is encoded as a polynomial and evaluated at n points (one per committee member). A vector commitment (using KZG) binds the evaluations to the polynomial commitment, producing a compact proof that each committee member received a correct evaluation. This provides *bribery resistance*: even if an adversary bribes a threshold of committee members, the algebraic binding prevents them from claiming to hold data they do not possess.
- **Mascarpone (certification layer):** Committee members who receive and verify their evaluations produce threshold signatures attesting to availability. A DA certificate is formed when $\geq 2/3$ of the committee has signed, providing a compact ($O(1)$-size) availability attestation.
- **Ladyfinger (fallback layer):** A secondary retrieval path through a CDN-like infrastructure that provides availability even if committee members go offline, with cryptographic proofs linking the fallback data to the original VID commitments.

Espresso's design provides *deterministic* confidence in availability: there is no probabilistic sampling, and finalization is conditional on the DA certificate. The Savoiardi VID layer's bribery-resistance guarantees go beyond what DAS provides. While DAS detects withholding probabilistically, Savoiardi's algebraic binding makes it infeasible for a committee member to falsely attest to holding data.

The principal limitation is that data retrieval requires cooperation from the committee (or the Ladyfinger fallback). There is no mechanism for an independent light client to sample data and verify availability without relying on the committee's attestations. Espresso explicitly argues that DAS introduces "unnecessary redundancies" given the algebraic guarantees provided by VID [6], but this design choice means that trust in data retrievability ultimately rests on the committee and fallback infrastructure.

4) Cross-Cutting Finding: No Slashing for Data Withholding: Across all six protocols, *none implements slashing for pure data withholding*. While each protocol provides mechanisms to detect or discourage withholding, the enforcement is uniformly indirect:

- **Polkadot:** Validators are slashed for casting invalid *validity* votes (attesting that an invalid parachain block is valid), but there is no slashing for failing to store or serve erasure-coded chunks. A validator that withholds its chunk simply does not sign the availability bitfield, potentially delaying but not corrupting finality [5].
- **Celestia:** Bad Encoding Fraud Proofs (BEFPs) allow any full node to prove that the block producer performed incorrect erasure coding [10]. However, BEFPs address encoding correctness, not data withholding. A block producer that correctly encodes data but subsequently withholds it faces no direct slashing penalty.
- **Ethereum:** PeerDAS enforces availability through peer scoring and disconnection: validators that fail to serve their custody columns receive negative gossip scores and may be disconnected from the p2p network. There is no protocol-level slashing for custody failures [2].
- **Espresso:** In the current Mainnet 0 deployment, enforcement is reputational only. No slashing mechanism is implemented for committee members that fail to serve their VID evaluations [6].
- **NEAR:** A challenge mechanism exists in the block structure that theoretically allows validators to dispute chunk unavailability, but this mechanism is "not used today" in the production network. Enforcement relies on the consensus protocol's liveness requirements [7].
- **Avail:** Slashing is implemented for equivocation (double-signing) but not for data withholding. Validators that fail to make sampled cells available face no direct economic penalty [8].

This universal absence of withholding-specific slashing reflects a fundamental difficulty: proving that a node *possesses* data but *refuses* to serve it is harder than proving that a node served *incorrect* data. Data withholding is an omission fault rather than a commission fault, and attributing omission to malice (rather than network partition, latency, or node failure) remains an open problem in distributed systems.

Table III consolidates the architectural comparison across all dimensions discussed in this section. Several cross-cutting observations emerge. First, DAS-capable protocols (Celestia, Avail, and the emerging Ethereum PeerDAS) represent a distinct design philosophy from validator-recovery protocols (Polkadot, NEAR) and VID-based designs (Espresso). The former enable trust-minimized light clients at the cost of network-level assumptions, while the latter two provide deterministic guarantees conditioned on committee or validator honesty. Second, the choice of commitment scheme is tightly coupled to the verification paradigm. KZG enables algebraic verification needed for DAS and VID, while Merkle-based schemes are sufficient for validator-recovery models where the validator set itself performs encoding verification. Third, all protocols share the same $f < n/3$ Byzantine threshold (the classical BFT bound) but differ substantially in how this threshold manifests in practice: Polkadot gates finality, Ethereum distributes custody, Celestia relies on sampling participation, Espresso requires committee certification, NEAR enforces per-shard honesty, and Avail combines sampling with algebraic commitments. These architectural choices have direct empirical consequences for throughput, cost, and operational characteristics, which we measure in Sections IV and V.

IV. Methodology

This section details our data collection framework, metric definitions, normalization approach, and reproducibility provisions.

TABLE III: Comprehensive Feature Comparison of DA Protocols

Feature	Polkadot	Ethereum	Celestia	Espresso	NEAR	Avail
Encoding	1D RS (validators)	1D RS (PeerDAS)	2D RS ($k \to 2k$)	Poly. VID	1D RS (chunks)	1D RS (rows)
Commitment	Merkle trie	KZG	NMT	KZG + vector	Merkle	KZG
Redundancy	$\sim 3\times$	$\sim 2\times$	$\sim 4\times$	Configurable	$\sim 3\times$	$\sim 2\times$
DAS support	No	Emerging (PeerDAS)	Yes (2D)	No (VID)	No	Yes (1D)
Light client DAS	No	Validator custody	Yes (16 samples)	No	No	Yes (10 samples)
Block time	6 s (relay)	12 s	6 s	~ 2 s	~ 0.6 s	20 s
Max DA capacity	10 MiB/core $\times$ 100	2.63 MiB (BPO2)	~ 8 MiB	~ 1 MiB	4 MiB/shard $\times$ 9	~ 4 MiB
Finality	GRANDPA ($\sim$12–60 s)	Casper FFG ($\sim$13 min)	Single-slot ($\sim$6 s)	HotShot ($\sim$1 s)	~ 2 s (Doomslug)	GRANDPA ($\sim$20–6
DA retention	~ 24 h	~ 18 days	~ 30 days	7 d	~ 5 epochs	~ 85 min
Fee model	Weight-based	EIP-1559 blob market	Gas-based	Per-byte	Gas-based	Weight-based
Withholding	Indirect (finality gate)	Peer scoring	BEFPs (encoding)	Reputational	Challenge (unused)	None (equivoc. on
Byzantine threshold	$f < n/3$	$f < n/3$	$f < n/3$	$f < n/3$	$f < n/3$ per shard	$f < n/3$

A. Data Collection Framework

We collect per-block production data from six DA protocols over a 90-day observation window (November 15, 2025 through February 14, 2026). For each protocol, we develop custom collection scripts that interface with mainnet RPC endpoints or explorer APIs, extracting block-level metrics at the granularity needed for throughput and cost analysis. All collectors are resumable: interrupted runs checkpoint progress in per-day CSV files and resume from the last collected block.

Table IV summarizes the data sources, block counts, and collection methods for each protocol.

For each protocol, the collector captures the following:

a) Polkadot: We scan relay chain blocks via the Polkadot.js RPC API, extracting three event types per block: `paraInclusion.CandidateBacked`, `paraInclusion.CandidateIncluded`, and `paraInclusion.CandidateTimedOut`. We also parse the `paraInherent.enter` extrinsic to extract signed availability bitfield counts and per-core availability fractions. Throughput is computed as an upper bound: included $\times$ maxPovBytes/cadence, since the relay chain does not expose actual PoV sizes (only commitments). Cost data is collected separately from the Coretime chain, scanning `broker.Purchased`, `broker.Renewed`, and `broker.SaleInitialized` events, as well as relay chain `onDemand.OnDemandOrderPlaced` events.

b) Ethereum: We collect per-block header fields (`blob_gas_used`, `excess_blob_gas`) from execution-layer RPC endpoints across the full 90-day window, which spans three distinct parameter regimes: Pectra (target 6, max 9), BPO1 (target 10, max 15, activated December 9, 2025), and BPO2 (target 14, max 21, activated January 7, 2026). The blob base fee is computed using the EIP-4844 exponential pricing formula. ETH/USD prices are sampled hourly from CoinGecko.

c) Celestia: Block data is retrieved from the Celenium indexer API, which provides 22 columns per block including `blobs_size`, `blobs_count`, `square_size`, `fill_rate`, and `fee_utia`. The observation window spans two protocol eras: the Ginger era (app versions 3–5, 64×64 max square, 6s blocks) and the Current era (app version 6+, 128×128 max square, 8 MiB capacity), with the transition occurring within our collection period.

d) Espresso: We fetch all blocks from the Espresso query-service explorer API using binary search to locate the start height and 20 concurrent workers fetching 100 blocks per page ($\sim$1,630 blocks/s collection rate). Each block record contains `size_bytes`, `num_transactions`, and `block_time_ms`. Since HotShot uses responsive consensus, block times are variable (no fixed cadence). The base fee has been verified as 1 wei/byte since genesis across headers from height 1K to 10.3M.

e) NEAR: We employ a dual-source strategy: NEAR Lake S3 for bulk historical collection ($\sim$275 blocks/s at concurrency 100) and NEAR RPC as fallback ($\sim$2.5 blocks/s). Per-block fields include `num_shards`, `total_encoded_bytes` (sum of chunk encoded lengths), `total_gas_used`, and `gas_price`.

f) Avail: The collector uses the Polkadot.js API (Avail is Substrate-based) to extract `submit_data_bytes`, `submit_data_count`, and `block_fee_plancks` (from `TransactionFeePaid` events) per block. Runtime version (`spec_version`) is tracked to account for parameter changes.

B. Metric Definitions

We define four primary metric categories, applied consistently across all protocols.

1) Throughput (MiB/s): We distinguish three throughput measures:

- **Protocol maximum**: the theoretical ceiling given current parameters. Protocol-specific formulas:

$$T_{\max}^{\text{Polkadot}} = \frac{\text{effective\_cores} \times \text{max\_pov}}{\text{cadence}} \tag{1}$$

$$T_{\max}^{\text{Ethereum}} = \frac{\text{max\_blobs} \times 128 \text{ KiB}}{\text{slot\_time}} \tag{2}$$

$$T_{\max}^{\text{Celestia}} = \frac{\text{max\_data\_bytes}}{\text{avg\_block\_time}} \tag{3}$$

TABLE IV: Data Collection Summary Across Six DA Protocols (90-Day Window)

Protocol	Source	Blocks	Key Fields	Language	Price Source
Polkadot	Relay chain RPC	∼1.3M	candidates backed/included, bitfields, core index	TypeScript	CoinGecko (DOT)
Ethereum	Execution + Beacon RPC	∼648K	blob_gas_used, excess_blob_gas, blob count	Python/SQL	CoinGecko (ETH)
Celestia	Celenium API	∼1.3M	blobs_size, square_size, fill_rate, fee_utia	Python	CoinGecko (TIA)
Espresso	Query-service API	∼2.9M	size_bytes, num_transactions, block_time_ms	Python	CoinGecko (ETH)
NEAR	NEAR Lake S3 + RPC	∼7.8M	total_encoded_bytes, gas_used, gas_price	TypeScript	CoinGecko (NEAR)
Avail	Substrate RPC	∼389K	submit_data_bytes, block_fee_plancks	TypeScript	CoinGecko (AVAIL)

where effective_cores $= \min(\text{num\_cores}, \lfloor \text{validators}/5 \rfloor)$ and cadence is 6s (async backing) for Polkadot.

- **Observed throughput**: measured from production data as

$$T_{\text{obs}} = \frac{\sum \text{payload\_bytes}}{\Delta t \times 2^{20}} \quad \text{(MiB/s)} \tag{4}$$

aggregated at hourly and daily granularity.
- **Utilization**: $U = T_{\text{obs}}/T_{\max}$, representing the fraction of available DA capacity in use.

2) Cost ($/MiB): Cost normalization requires care, as protocols use fundamentally different pricing models:

- **Per-byte fee markets** (Celestia, Espresso, Avail): cost is directly computable from on-chain fee data and token prices.

$$\text{cost\_per\_MiB} = \text{fee\_per\_byte} \times 2^{20} \times \text{token\_USD} \tag{5}$$

- **Blob gas market** (Ethereum): cost follows EIP-4844 exponential pricing:

$$\text{cost\_per\_MiB} = \frac{\text{baseFeePerBlobGas} \times 131{,}072}{10^9} \times \text{ETH\_USD} \times \frac{2^{20}}{131{,}072} \tag{6}$$

- **Coretime pricing** (Polkadot): DA is bundled with execution capacity. We compute:

$$\text{cost\_per\_MiB} = \frac{\text{coretime\_price\_DOT} \times \text{DOT\_USD}}{\text{max\_MiB\_per\_region}} \tag{7}$$

for bulk coretime, and fee_paid/max_pov_MiB for on-demand. This yields a *best-case* cost assuming full PoV utilization every block.
- **Gas + storage staking** (NEAR): upload cost is gas-based (burned); state storage requires locking 1 NEAR per 100 KB. Rollup DA blobs avoid state costs by using function call inputs (pruned).

We report spot cost at percentiles (p50, p90, p99) and volume-weighted average price (VWAP) over the 90-day window.

3) Annualized Cost: Since DA is universally temporary (all protocols prune data), continuous availability requires periodic reposting. We define:

$$\text{cost\_per\_MiB\_year} = \text{cost\_per\_MiB} \times \left\lceil \frac{365}{\text{retention\_days}} \right\rceil \tag{8}$$

This normalization is essential for fair comparison, as short-retention protocols (Avail: ∼85 min, Polkadot: 25 hours) appear cheap at spot but expensive when annualized.

4) Block Dynamics: We measure block time distributions (p10, p50, p90), block size distributions, and empty block rates. For protocols with variable block times (Espresso's responsive HotShot consensus), we report the empirical distribution rather than a fixed value.

C. Normalization and Comparability

Several methodological challenges arise from comparing heterogeneous protocols:

1) **Demand-constrained measurements**: all six protocols operate well below their theoretical capacity during our observation window (<50% utilization). Observed throughput therefore reflects *demand*, not protocol capability. We report both observed and maximum throughput and emphasize that empirical results characterize current usage patterns, not protocol limits.
2) **Token price volatility**: USD-denominated costs are sensitive to token prices, which fluctuated during our window (e.g., ETH ranged $2,000–$4,000; TIA ranged $0.30–$0.85). We use contemporaneous hourly prices from CoinGecko rather than fixed spot rates, and report VWAP alongside percentile distributions.
3) **Apples-to-oranges pricing**: Polkadot prices blockspace (coretime), not bytes. Ethereum prices blob gas. Celestia and Espresso price per byte. We normalize all to $/MiB but note that Polkadot's cost includes execution capacity (not just DA), making its per-MiB figure a lower bound on pure DA cost.
4) **Upper-bound throughput for Polkadot**: the relay chain does not expose actual PoV sizes on-chain (only commitments). Our throughput measurement uses included_candidates × max_pov_size, which is an upper bound. Actual throughput is lower since real parachains typically use 1–3 MiB per candidate.
5) **Protocol upgrades within window**: Ethereum underwent two parameter changes (BPO1, BPO2) and Celestia transitioned eras during our observation period. We segment analysis by protocol era where applicable and treat these transitions as natural experiments (Section V-F).
6) **Throughput measurement conventions**: Two definitional choices affect comparability with other sources (e.g., L2Beat). First, for Ethereum we report protocol maximum throughput using the *absolute maximum* blob count per block (21 under BPO2), whereas some trackers use the EIP-4844 *target* blob count (14 under BPO2), below which fees trend toward zero. Both are valid: the

target represents the fee-market equilibrium, while the absolute maximum represents the hard protocol ceiling. Second, for Celestia we measure *user data throughput* (the original data square, ODS), not the full 2D Reed–Solomon encoded square. The extended square is $4\times$ larger (a $k\times k$ ODS becomes $2k\times 2k$ after encoding), so sources that report total encoded bandwidth will show approximately $4\times$ our figures. We report user data throughput because it reflects the amount of application-level data that can be posted per unit time.

D. Reproducibility

All collection scripts, raw datasets, analysis notebooks, and generated figures are published in a public repository under Apache 2.0 (code) and CC BY 4.0 (data) licenses. Docker environments encapsulate dependencies for each collector. Per-protocol `chain_config.json` files record exact protocol parameters, governance milestones, and era boundaries active during collection. Analysis is implemented in Python (pandas, matplotlib) with Jupyter notebooks that regenerate all figures and tables from the raw CSV data. Additionally, interactive dashboards are published for real-time exploration of the collected data.

V. Empirical Results

This section presents our empirical findings from 90 days of production data across all six DA protocols, organized by metric category.

A. Observation Window and Protocol Eras

Our observation window (November 15, 2025 through February 14, 2026) captured several protocol parameter changes that serve as natural experiments:

- **Ethereum**: three distinct regimes: Pectra (target 6/max 9 blobs), BPO1 (target 10/max 15, from Dec 9), and BPO2 (target 14/max 21, from Jan 7).
- **Celestia**: transition from Ginger era (64×64 max square, 2 MiB capacity) to Current era (128×128 max square, 8 MiB capacity).
- **Polkadot**: stable at 100 cores with 10 MiB max PoV throughout the window.
- **Espresso, NEAR, Avail**: no major parameter changes during the window.

Table V summarizes the active protocol parameters during the observation period.

B. Throughput and Utilization

1) Observed vs. Maximum Throughput: Figure 1 presents per-protocol throughput time series, plotting observed throughput (actual DA payload per unit time) against the protocol maximum. All protocols use a logarithmic y-axis to accommodate the wide dynamic range between observed and maximum throughput.

Key observations:

1) **Universal under-utilization**: No protocol sustained more than 50% utilization over the 90-day window. Ethereum showed the highest sustained utilization ($\sim$60–70% during the Pectra era, declining to $\sim$20–30% after each BPO capacity increase). Celestia operated at $\sim$3–5% in steady state, with a single extraordinary spike exceeding 100% utilization during December 13–22. Espresso peaked at $\sim$3% utilization and Avail at $\sim$3–5%.
2) **Demand, not supply**: Because all protocols are demand-constrained, observed throughput reflects ecosystem adoption and rollup activity, not protocol performance limits. Comparing raw observed MiB/s across protocols would be misleading (it would measure market share, not engineering capability).
3) **Polkadot's upper-bound caveat**: Our Polkadot throughput uses included × max_pov, yielding an upper bound. Actual PoV sizes are smaller, so true throughput is lower than reported.

2) Utilization: Figure 2 shows DA utilization as a percentage of protocol capacity.

Table VI provides aggregate throughput and utilization statistics.

C. Cost Analysis

1) Spot Cost ($/MiB): Figure 3 presents per-protocol cost quantile bands (p10/p50/p90) over the observation window. The y-axes use logarithmic scaling, which is essential given the seven-order-of-magnitude range across protocols.

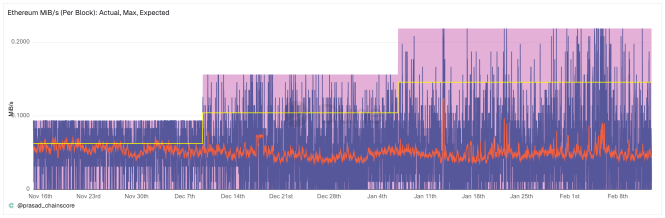
2) Cross-Protocol Cost Comparison: Table VII presents the cost distribution across protocols.

Key findings:

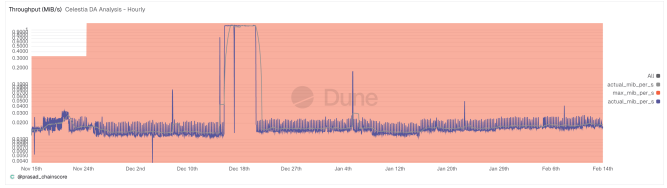
1) **Seven orders of magnitude**: DA costs range from $\sim\$3\times10^{-9}$/MiB (Espresso) to $\sim$\$0.80/MiB (Avail and Celestia p99 peaks). This range reflects different economic designs, not just protocol efficiency.
2) **Espresso's cost is not economically meaningful**: At 1 wei/byte fixed since genesis, Espresso's DA fee is a placeholder that has never been adjusted. The cost curve (Figure 3c) directly mirrors ETH/USD price fluctuations, declining from $\sim\$3.5\times10^{-9}$ to $\sim\$2\times10^{-9}$ as ETH price fell from $\sim$\$3,500 to $\sim$\$2,600 during our window. This is a permissioned Mainnet 0 artifact, not a production fee market.
3) **Avail exhibits the widest cost spread**: While Avail's p50 cost (\$0.008/MiB) is comparable to Celestia's, its p99 spikes to $\sim$\$0.80 (matching Celestia's peak) despite much lower utilization. These spikes coincide with brief activity surges around December 4 and late December, suggesting the fee modifier mechanism amplifies cost volatility even at low absolute throughput.
4) **Ethereum's Pectra-era pricing anomaly**: During the Pectra era (pre-Dec 9), Ethereum's p50 cost was $\sim\$4\times10^{-10}$/MiB, comparable to Espresso's placeholder fee. The BPO capacity increases paradoxically *increased* observed median cost by attracting sufficient demand to activate the exponential pricing mechanism.
5) **Celestia's cost is trending downward**: Over the full window, Celestia's cost declined from $\sim$\$0.05 to

TABLE V: Active Protocol Parameters During 90-Day Observation Window

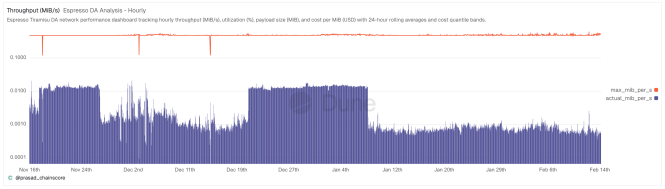
Parameter	Polkadot	Ethereum	Celestia	Espresso	NEAR	Avail
Block/slot time	6 s	12 s	6 s	∼2–3 s	∼0.6 s	20 s
Max DA per block	10 MiB × cores	2.63 MiB (BPO2)	8 MiB	∼1 MiB	4 MiB × 9 shards	∼4 MiB
Protocol max MiB/s	∼167	∼0.22	∼1.31	∼0.37	∼59	∼0.21
Retention	25 h	∼18 d	30 d	7 d	∼2.5 d	∼85 min
Native token	DOT	ETH	TIA	ETH (fees)	NEAR	AVAIL



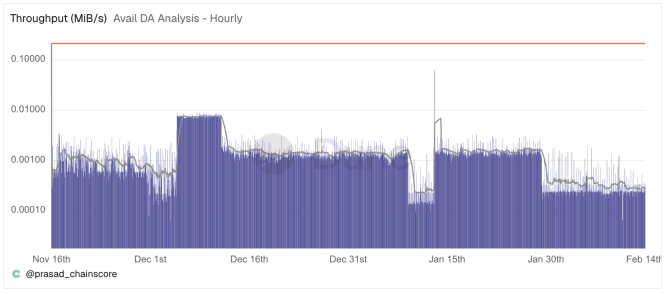
(a) Ethereum



(b) Celestia



(c) Espresso



(d) Avail

Fig. 1: Observed DA throughput (MiB/s) versus protocol maximum for four protocols over the 90-day observation window (Polkadot and NEAR omitted pending data collection). All protocols operate well below their theoretical capacity ceiling (red/shaded region), indicating demand-constrained rather than supply-constrained regimes. Ethereum (a) shows clear step-function increases in the maximum at BPO1 (Dec 9) and BPO2 (Jan 7). Celestia (b) shows the December 13–22 throughput surge approaching the protocol maximum. Espresso (c) exhibits two distinct activity phases (Nov, Dec–Jan) followed by collapse. Avail (d) shows peak activity around Dec 4–10 with a subsequent decline.

TABLE VI: 90-Day Throughput and Utilization Summary

Protocol	Max MiB/s	p50 MiB/s	Mean MiB/s	Mean Util.
Polkadot	∼167	TBD	TBD	TBD
Ethereum	0.22	∼0.045	∼0.051	∼23%
Celestia	1.31	∼0.018	∼0.024	∼5%
Espresso	0.37	∼0.002	∼0.003	∼1.5%
NEAR	∼59	TBD	TBD	TBD
Avail	0.21	∼0.001	∼0.002	<1%

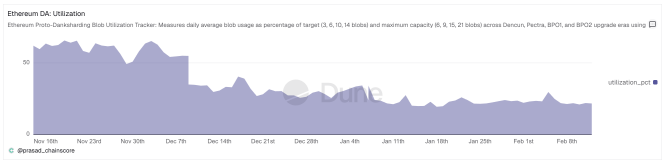
TABLE VII: 90-Day Cost Summary ($/MiB)

Protocol	p50	p90	p99	VWAP	Pricing
Polkadot	TBD	TBD	TBD	TBD	Coretime
Ethereum	$\sim 4\times10^{-4}$	$\sim 5\times10^{-3}$	∼0.021	$\sim 5\times10^{-3}$	Blob gas
Celestia	∼0.021	∼0.06	∼0.40	∼0.03	Gas/byte
Espresso	$\sim 3\times10^{-9}$	$\sim 3.5\times10^{-9}$	$\sim 8\times10^{-9}$	$\sim 3\times10^{-9}$	Fixed fee
NEAR	TBD	TBD	TBD	TBD	Gas
Avail	∼0.008	∼0.03	∼0.80	∼0.02	Fee+modifier

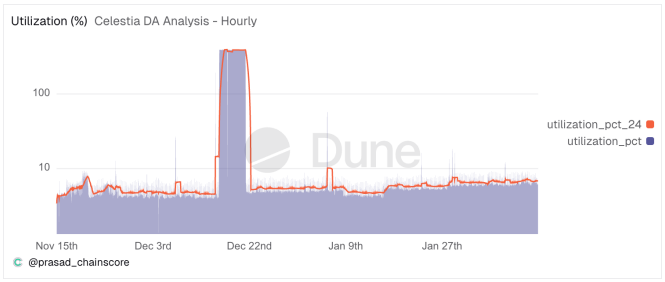
∼$0.02/MiB (Figure 3b), partially reflecting TIA token price decline ($0.85→$0.30) and partially reflecting post-upgrade capacity growth.

3) Annualized Cost: Table VIII normalizes cost by retention window, revealing the true cost of continuous data availability.

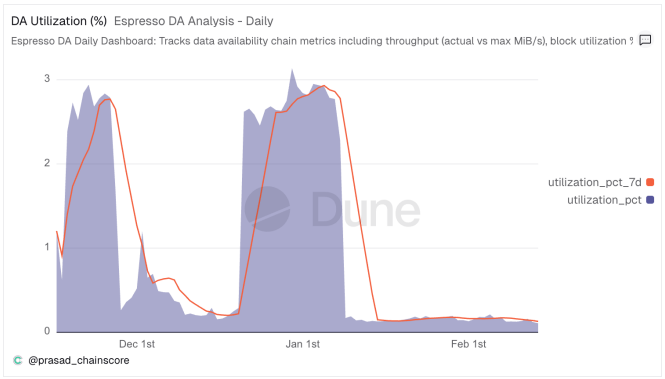
This normalization reveals that protocols with short retention windows that appear inexpensive at spot cost may become expensive when continuous availability is required. Avail's 85-minute retention means data must be reposted ∼6,176 times per year, amplifying its p50 cost from a modest $0.008/MiB to ∼$49/MiB/year—the most expensive option by this measure. Conversely, Ethereum's relatively long 18-day retention means its annualized multiplier is only ∼21×, and Celestia's 30-day



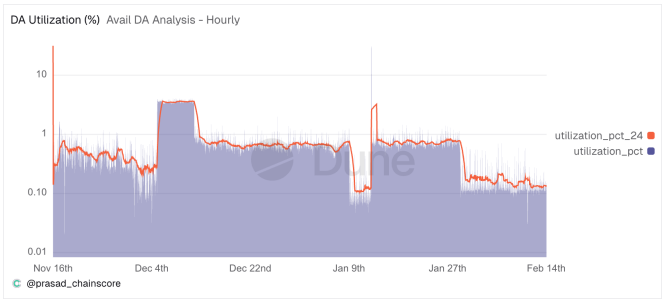
(a) Ethereum



(b) Celestia



(c) Espresso



(d) Avail

Fig. 2: DA utilization (%) for four protocols over the 90-day observation window (Polkadot and NEAR pending). Ethereum (a) shows the highest sustained utilization (∼65% in Pectra), with clear step-drops after each BPO capacity increase as the denominator grew faster than demand. Celestia (b) experienced the only near-saturation event: utilization exceeded 100% during December 13–22, driven by a sustained demand surge. Espresso (c) shows two usage phases peaking at ∼3%, with near-complete collapse by February. Avail (d) typically operates at 0.1–1% utilization with brief spikes to ∼3–5%. All panels use logarithmic y-axes except Ethereum.

TABLE VIII: Annualized DA Cost ($/MiB/year), Accounting for Retention

Protocol	Retention	Reposts/yr	p50 $/MiB/yr
Avail	∼85 min	∼6,176	∼49.4
Polkadot	25 h	∼350	TBD
NEAR	∼2.5 d	∼146	TBD
Espresso	7 d	∼52	$\sim 1.6\times10^{-7}$
Ethereum	∼18 d	∼21	∼0.008
Celestia	30 d	∼12	∼0.25

window yields a 12× multiplier.

D. Block Dynamics

Figure 4 shows block time distributions for Celestia and Espresso (the two protocols where block time dynamics are most informative). Fixed-cadence protocols (Ethereum 12 s, Polkadot 6 s, Avail 20 s) produce trivially tight block time distributions by design.

E. Daily Payload Volume

Figure 5 presents daily DA payload volume for each protocol, providing an absolute measure of ecosystem adoption that complements the rate-normalized throughput metric.

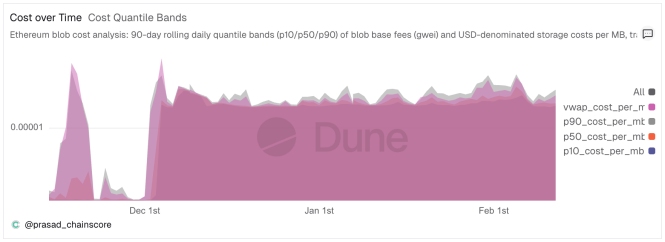
Espresso's payload pattern warrants specific discussion. The two distinct activity phases (mid-November and late December through early January), followed by near-complete collapse to <50 MiB/day, suggest non-organic traffic patterns characteristic of a permissioned Mainnet 0 deployment in its early stages. This contrasts with Celestia's more sustained baseline activity (30–100 MiB/hr even outside the December surge), which reflects organic rollup usage.

F. Natural Experiments: Protocol Upgrades

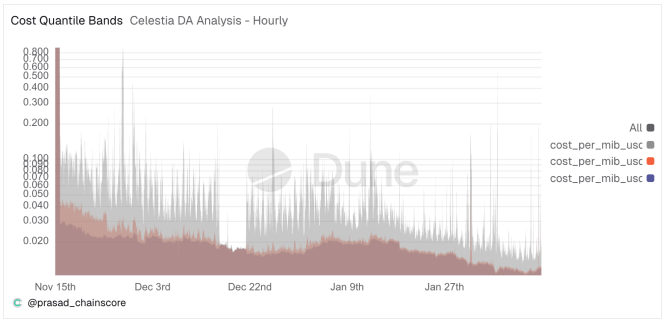
Two protocol transitions within our observation window provide natural experiments for studying how parameter changes affect DA markets.

1) Ethereum: Pectra → BPO1 → BPO2: Ethereum's blob capacity increased in two steps during our window: from target 6/max 9 (Pectra) to target 10/max 15 (BPO1, Dec 9) to target 14/max 21 (BPO2, Jan 7). Each increase created a capacity shock visible across multiple metrics.

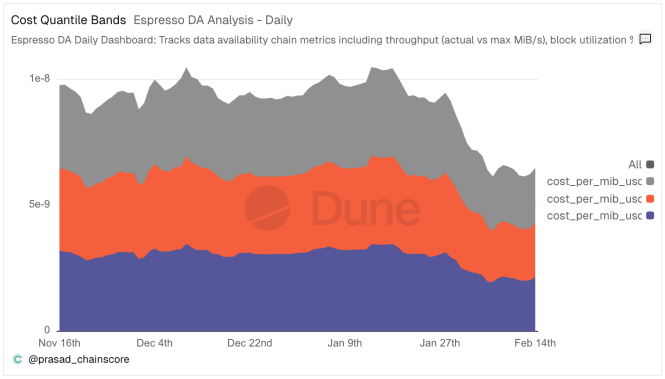
- **Utilization drop**: each capacity increase temporarily reduced utilization (Figure 2a), as the denominator (max blobs) grew faster than demand. Utilization fell from



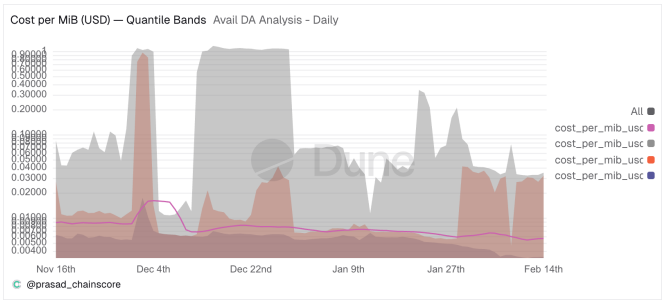
(a) Ethereum



(b) Celestia



(c) Espresso



(d) Avail

Fig. 3: Cost quantile bands ($/MiB) for each protocol, showing fee market maturity and price volatility. Ethereum (a) shows an initial spike during the Pectra era followed by stabilization at ∼$0.00001 under BPO1/BPO2. Celestia (b) exhibits a downward cost trend from ∼$0.05 to ∼$0.02 over the window, partially driven by TIA price decline. Espresso (c) has extremely tight bands (p10/p50/p90 nearly identical) in the $\sim\$3\times10^{-9}$ range, tracking ETH/USD since the 1 wei/byte fee is fixed. Avail (d) shows the widest spread: p50 at ∼$0.008 but p99 spikes reaching $0.80+, indicating the most volatile cost distribution among the four.

∼65% (Pectra) to ∼25% (post-BPO1) to ∼15% (post-BPO2).

- **Fee regime shift**: the exponential pricing mechanism responded dramatically. Figure 3a shows the Pectra era had a brief initial cost spike followed by near-zero costs (p50 $\sim\$4\times10^{-10}$), while BPO1/BPO2 stabilized at a *higher* median cost ($\sim\$4\times10^{-4}$) as increased capacity attracted more demand.
- **Gradual recovery**: utilization partially recovered over subsequent weeks as lower fees attracted additional demand, though it did not return to Pectra-era levels.

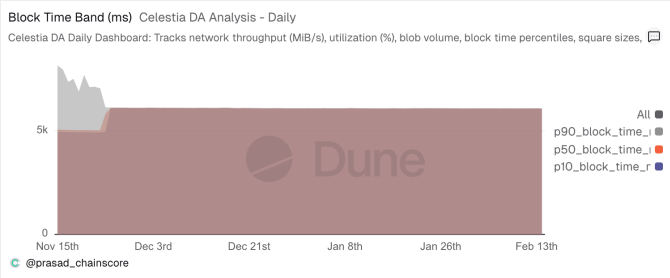
2) Celestia: December Utilization Spike: During December 13–22, Celestia experienced a sustained demand surge that pushed utilization near and beyond 100%. This was the only near-saturation event observed across any protocol in our window (Figure 2b).

- **Throughput surge**: hourly payload volume spiked from a baseline of 30–100 MiB/hr to over 1,000 MiB/hr (Figure 5a), with observed throughput approaching the protocol maximum of ∼1.31 MiB/s (Figure 1b).
- **Fee market response**: cost per MiB increased during the event, with p90 costs reaching ∼$0.10–$0.20, demonstrating that Celestia's gas pricing mechanism responds to congestion.
- **Post-event recovery**: utilization returned to <10% after the event, costs reverted toward the declining baseline trend, and hourly payload volume returned to the 30–60 MiB range.

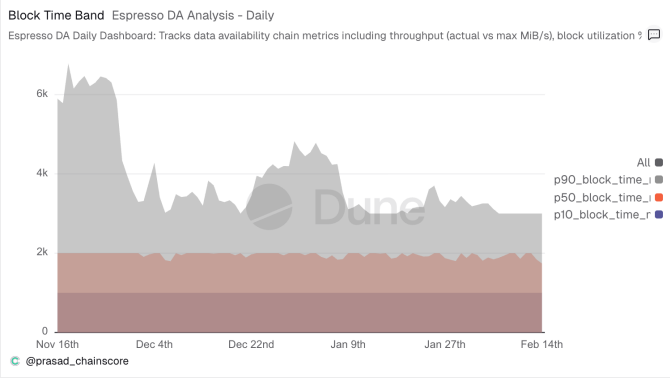
This event provides empirical evidence that Celestia's fee market functions as designed under load, a validation that is typically only demonstrated through synthetic stress tests. The event also confirms our characterization of the DA landscape as demand-constrained: even Celestia's saturation was a transient event lasting approximately 9 days, after which the protocol returned to steady-state under-utilization.

VI. Security and Trust Analysis

This section analyzes the security properties and trust assumptions of each DA protocol, focusing on Byzantine fault tolerance, data withholding enforcement, retention guarantees, and decentralization characteristics.



(a) Celestia



(b) Espresso

Fig. 4: Block time distributions (p10/p50/p90 bands) for Celestia and Espresso. Celestia (a) shows tight bands around 5–6 s after an initial elevated p90 in mid-November, demonstrating stable CometBFT fixed-cadence consensus. Espresso (b) shows characteristic responsive HotShot consensus behavior: p10 $\sim$1 s, p50 $\sim$2 s, and a highly variable p90 ranging from 2 to 7 s. The p90 variability in Espresso indicates periods of consensus pressure, particularly in the early weeks (p90 $\sim$6.5 s) that gradually stabilized toward $\sim$3 s by February.

A. Byzantine Fault Tolerance

All six protocols inherit the classical BFT bound of $f < n/3$, where f is the number of Byzantine-faulty nodes and n is the total committee or validator set size. However, the practical manifestation of this threshold varies substantially.

a) Polkadot: Polkadot's relay chain requires $\geq 2/3$ of validators to sign availability bitfields before a parachain block can be included and finalized. With approximately 600 active validators, the system tolerates up to $\sim$200 Byzantine faults. The ELVES protocol further strengthens this through approval checking: randomly selected validators re-execute parachain blocks post-inclusion, and a single invalid execution triggers a dispute that can slash the responsible validators [5]. The probability of an invalid block escaping detection decreases exponentially with the number of approval checkers.

b) Ethereum: Ethereum's Casper FFG requires a $2/3$ supermajority of validators (by stake weight) to finalize blocks. With approximately 1 million active validators, the economic security backing finality exceeds $30 billion (at ETH $\sim$$3,000). PeerDAS distributes custody across the validator set: each validator holds a subset of blob columns, and system-level availability confidence emerges from the aggregate custody distribution [3]. The large validator set makes coordinated withholding attacks economically prohibitive.

c) Celestia: Celestia runs CometBFT (Tendermint) consensus, which requires $\geq 2/3$ of validators (by stake weight) to commit each block. The validator set is capped at 100 in the current configuration. DAS provides an additional layer of security: even if validators finalize a block with unavailable data, light client sampling will probabilistically detect the withholding [10]. This two-layer model (consensus BFT + sampling) provides stronger guarantees than consensus alone, at the cost of requiring sufficient light client participation.

d) Espresso: Espresso's HotShot consensus operates with 100 nodes across 20 operators in the Mainnet 0 configuration. The DA certificate requires $\geq 2/3$ of committee signatures, tolerating up to $\sim$33 Byzantine nodes. However, the small operator count (20) means that the effective trust assumption is closer to $f < 7$ at the operator level [6].

e) NEAR: NEAR's Nightshade sharding applies the BFT threshold per shard. With 9 shards and approximately 400 validators, the per-shard committee size depends on stake distribution. A critical assumption is that each shard independently maintains $< 1/3$ Byzantine nodes; an adversary who concentrates stake in a single shard can potentially compromise that shard's DA at lower total cost than compromising the network-wide threshold [7]. In practice, however, most mainnet validators currently track all shards rather than only their assigned shard, which means the per-shard BFT assumption is more theoretical than operational: DA benefits from full-network replication even though the protocol is designed for shard-local storage.

f) Avail: Avail's BABE/GRANDPA consensus requires $\geq 2/3$ honest validators for both block production and finality. DAS light clients provide independent verification, analogous to Celestia's model. The VectorX bridge to Ethereum introduces an additional trust surface: bridge liveness depends on the prover infrastructure, and the approximately 2-hour bridge delay creates a window during which Avail-native finality and Ethereum-bridged attestations may diverge [8].

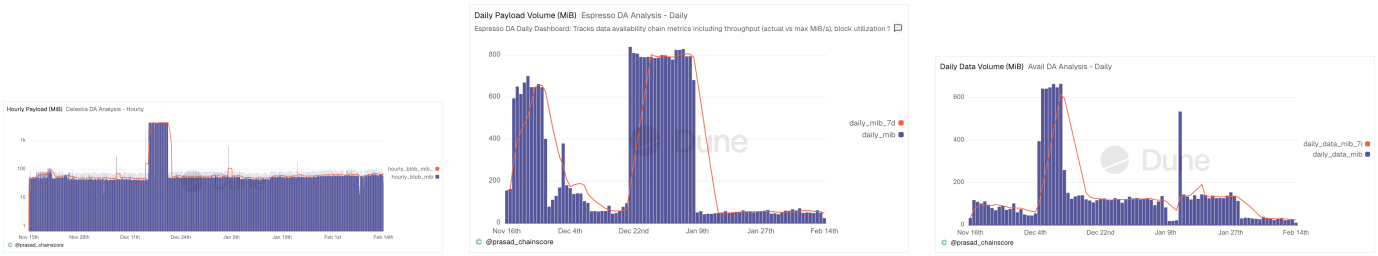
Table IX summarizes the BFT characteristics across protocols.

B. Data Withholding Enforcement

As established in Section III-C, no protocol implements slashing specifically for data withholding. Here we analyze the indirect enforcement mechanisms and their practical effectiveness.

a) Detection mechanisms: Protocols employ three distinct detection approaches:

1) **Sampling-based detection** (Celestia, Avail, Ethereum/PeerDAS): Light clients randomly request coded pieces; failure to receive responses indicates potential withholding. Detection probability increases with sampling participation. This approach requires an active network of sampling nodes and is vulnerable to targeted eclipsing attacks.
2) **Consensus-integrated attestation** (Polkadot, NEAR): Validators attest to chunk availability through protocol-defined messages (Polkadot's availability bitfields, NEAR's chunk endorsements). Detection



(a) Celestia **(b)** Espresso **(c)** Avail

Fig. 5: Daily DA payload volume (MiB) for Celestia, Espresso, and Avail over the 90-day window. Celestia (a) shows baseline hourly payload of 30–100 MiB, with the December 13–22 surge reaching >1,000 MiB/hr. Espresso (b) exhibits two burst periods (Nov: 500–700 MiB/day, Dec–Jan: 700–850 MiB/day) followed by collapse to <50 MiB/day—characteristic of a Mainnet 0 deployment with non-organic traffic. Avail (c) peaked at ∼650 MiB/day around Dec 4–10 with a similar declining trend.

TABLE IX: Byzantine Fault Tolerance and Validator Set Characteristics

Protocol	BFT Threshold	Validator Count	Consensus	Finality Latency	Staking Requirement
Polkadot	$f < n/3$	∼600	BABE + GRANDPA	12–60 s	∼1.8M DOT min
Ethereum	$f < n/3$ (stake)	∼1M	Casper FFG	∼12.8 min	32 ETH per validator
Celestia	$f < n/3$ (stake)	∼100	CometBFT	Single-slot (∼6 s)	Variable
Espresso	$f < n/3$	100 (20 operators)	HotShot	∼1–2 s	None (permissioned)
NEAR	$f < n/3$ per shard	∼400	Nightshade	∼2 s	Seat-price auction
Avail	$f < n/3$	∼100	BABE + GRANDPA	20–60 s	Variable

is deterministic—a missing attestation is visible on-chain—but only detects validator-level withholding, not targeted withholding from specific requesters.

3) **Committee certification** (Espresso): The DA certificate requires threshold signatures from committee members who have verified receipt of their VID shares. A missing certificate prevents finalization. Detection is deterministic but relies entirely on the committee's honest participation.

b) Enforcement gap: The fundamental challenge across all protocols is the *attribution gap*: detecting that data is unavailable does not identify the responsible party. A validator that fails to serve a chunk may be offline, network-partitioned, or maliciously withholding, and the protocol cannot distinguish these cases. This attribution difficulty explains why no protocol has implemented withholding-specific slashing: the false positive rate (slashing honest nodes experiencing network issues) would be unacceptable.

C. Retention and Persistence

DA is universally treated as temporary across all six protocols. Table X summarizes retention windows and the implications for downstream consumers.

The retention range spans two orders of magnitude, from Avail's ∼85 minutes to Celestia's ∼30 days. This variation has critical implications for rollup security:

- **Fraud proof windows**: Optimistic rollups require a challenge period (typically 7 days) during which fraud proofs can be submitted; the DA layer must retain data at least as long as the challenge period. Avail (85 min), Polkadot (25 h), and NEAR (2.5 d) all have retention windows shorter than the standard 7-day fraud proof window, requiring rollups to either archive data externally or implement shorter challenge periods.
- **Bridging latency**: Cross-chain bridges that verify DA attestations must complete verification within the retention window. Avail's VectorX bridge to Ethereum has an approximately 2-hour attestation delay, consuming a significant fraction of the 85-minute retention window (with race conditions in edge cases).
- **Archival burden**: After DA expiry, data persistence shifts to voluntary archival by full nodes, third-party indexers, or rollup operators themselves. No protocol guarantees post-retention data retrievability.

TABLE X: Data Retention Windows and Persistence Model

Protocol	Retention Window	Pruning Mechanism
Avail	∼85 min (256 blocks)	Substrate state pruning
Polkadot	∼25 h	Relay chain erasure pruning
NEAR	∼2.5 d (5 epochs)	Garbage collection
Espresso	1–7 d (configurable)	Query service retention
Ethereum	∼18 d (4,096 epochs)	Consensus-layer pruning
Celestia	∼30 d	Node storage rotation

D. Trust Assumptions Summary

Table XI maps each protocol's key trust assumptions.

TABLE XI: Trust Assumption Comparison Across DA Protocols

Protocol	Availability Assumption	Encoding Trust	Additional Assumptions
Polkadot	$\geq 2/3$ validators honest (bitfield voting)	Validators re-encode and verify (approval checking)	Relay chain liveness
Ethereum	$\geq 2/3$ validators honest (Casper FFG) + sufficient custody distribution	KZG algebraic verification (no fraud proofs needed)	Trusted setup integrity (KZG ceremony)
Celestia	$\geq 2/3$ validators honest + sufficient DAS participation	BEFPs from ≥ 1 honest full node	≥ 1 non-eclipsed honest full node
Espresso	$\geq 2/3$ committee signs DA certificate	KZG + vector commitment algebraic verification	Committee honesty; Ladyfinger fallback liveness
NEAR	$< 1/3$ Byzantine per shard	Chunk producers re-encode	Honest shard assignment; no stake concentration
Avail	$\geq 2/3$ validators honest + DAS participation	KZG algebraic verification	Trusted setup integrity; VectorX bridge liveness

The trust models fall into three categories. *DAS-enhanced protocols* (Celestia, Avail, Ethereum) provide the strongest light client guarantees but require active network participation in sampling. *Validator-recovery protocols* (Polkadot, NEAR) provide deterministic guarantees conditioned on validator honesty but offer no independent light client verification. *VID-based protocols* (Espresso) provide algebraic guarantees against bribery but depend on committee and fallback infrastructure availability.

VII. Discussion

This section synthesizes findings across the empirical and architectural analyses, discusses implications for protocol designers and rollup developers, and acknowledges limitations.

A. DAS vs. VID vs. Validator Recovery: Empirical Trade-offs

Our architectural taxonomy (Section III) identified three verification paradigms. The empirical data reveals how these paradigms manifest in practice:

a) DAS protocols (Celestia, Avail, Ethereum): These protocols offer the strongest trust-minimization for light clients; sampling enables resource-constrained nodes to independently verify DA. However, DAS protocols face the *participation problem*: sampling confidence degrades if insufficient nodes participate. Our observation of Celestia's December utilization spike demonstrates that DAS networks function correctly under load when participation is sufficient. Avail's short retention window (85 minutes) partially undermines DAS's trust-minimization advantage: the window for sampling is narrow, and late-joining light clients may be unable to verify historical data.

b) Validator-recovery protocols (Polkadot, NEAR): These protocols provide the tightest coupling between DA and consensus finality. Polkadot's finality-gating mechanism (GRANDPA will not finalize blocks with insufficient availability bitfields) is the most direct enforcement mechanism we observe: unavailable data literally cannot be finalized. The trade-off is the absence of independent light client verification; trust rests entirely on the validator set. For Polkadot's use case (parachain validation), this trade-off is acceptable since the relay chain's validators are the primary consumers of DA; for general-purpose DA (serving external rollups), this model requires rollup operators to trust the validator set.

c) VID protocol (Espresso): Espresso's Tiramisu provides algebraic bribery resistance (a property no other protocol offers), but in its current Mainnet 0 deployment, this theoretical advantage is undermined by the permissioned operator set (20 operators) and absence of slashing. The empirically observed near-zero cost (1 wei/byte) confirms that the fee market is a placeholder. As Espresso matures toward a permissionless deployment with active economic security, the VID paradigm's advantages may become more practically significant.

B. The Cost–Throughput Frontier

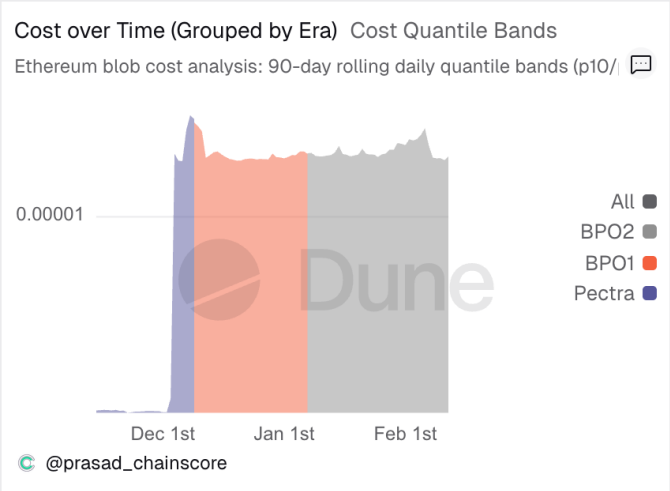
Plotting DA cost against protocol maximum throughput reveals no simple inverse relationship. Espresso offers both low cost and moderate throughput, but its cost is economically meaningless (Mainnet 0 artifact). Celestia and Avail provide moderate throughput at moderate cost. Polkadot provides the highest theoretical throughput ($\sim$167 MiB/s) at a cost that is difficult to compare directly since coretime prices bundled execution, not just DA. Ethereum provides the lowest per-block throughput among the six but benefits from the strongest network effects, economic security, and longest retention.

This observation underscores that *cost and throughput alone are insufficient metrics for DA evaluation*. Retention, trust assumptions, light client capabilities, and ecosystem integration are equally important, a finding that purely quantitative benchmarks risk overlooking.

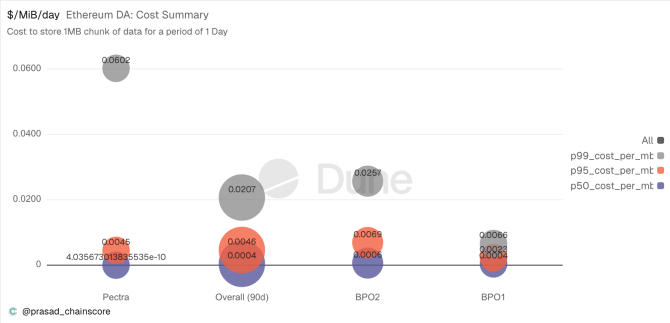
C. Implications for Rollup Developers

Our findings suggest several considerations for rollup developers selecting a DA backend:

1) **Retention must exceed fraud proof windows**: Optimistic rollups with 7-day challenge periods cannot safely use DA layers with shorter retention (Avail: 85 min, Polkadot: 25 h, NEAR: 2.5 d) without external archival infrastructure. This is a binding constraint that supersedes cost considerations.



(a) Cost quantile bands color-coded by protocol era



(b) Cost summary statistics ($/MiB/day) by era

Fig. 6: Ethereum DA cost segmented by parameter era. Panel (a) shows the cost quantile bands color-coded by Pectra (blue), BPO1 (orange), and BPO2 (gray) eras, illustrating the regime transitions. Panel (b) shows per-era cost percentile summaries: Pectra had the highest p99 ($0.060/MiB/day) but near-zero p50, while BPO1 and BPO2 show more moderate tails with higher medians. The BPO hardfork mechanism allows incremental DA scaling without full protocol upgrades.

2) **Spot cost is misleading without annualization**: A rollup evaluating DA cost should compute annualized cost (Section IV-B), which accounts for the number of times data must be reposted to maintain continuous availability. Short-retention protocols that appear cheap at spot may be expensive when annualized.
3) **Fee market maturity matters**: Our observation of Celestia's December congestion event shows that responsive fee markets produce cost spikes under load. Rollups sensitive to cost predictability should consider protocols with stable or capped pricing.
4) **Utilization headroom is currently abundant**: All protocols operate at $<$50% utilization. In the near term, capacity is not a differentiator—demand can be accommodated by any protocol. As adoption grows and utilization approaches saturation, the empirical dynamics we measure (fee market responsiveness, throughput scaling) will become increasingly important.

D. Protocol Design Observations

Several cross-cutting observations emerge from comparing six production protocols simultaneously:

a) The withholding enforcement gap is universal: No protocol has solved the fundamental problem of attributing data withholding to specific actors. This suggests that slashing for DA omission faults may be a harder problem than slashing for commission faults (equivocation, invalid execution). Future protocol designs may need to pursue alternative enforcement models: insurance-based schemes, reputation systems with gradual stake reduction, or cryptographic mechanisms that make withholding economically irrational rather than attributably punishable.

b) DA is converging toward ephemerality: All six protocols treat DA as temporary, with no protocol offering permanent data availability. This is a deliberate design choice reflecting the scalability–permanence trade-off: storing all historical data indefinitely does not scale, and DA's primary function is to provide a window for verification and fraud detection, not long-term archival. The DA layer is therefore not a substitute for archival storage; rollups and applications must implement separate data permanence strategies (e.g., IPFS, Filecoin, Arweave) for data that must be retrievable beyond the DA retention window.

c) Encoding redundancy vs. commitment sophistication: Protocols exhibit an inverse relationship between encoding redundancy and commitment scheme sophistication. High-redundancy encodings (Celestia's 4×, Polkadot/NEAR's 3×) use simpler commitment schemes (NMT, Merkle), while low-redundancy encodings (Ethereum/Avail's 2×) use algebraically richer commitments (KZG) that enable encoding correctness verification without fraud proofs. Espresso's polynomial VID represents a third point in this design space, using algebraic commitments to provide both encoding verification and bribery resistance with configurable redundancy.

E. Limitations and Threats to Validity

a) Demand-constrained measurements: All observed throughput figures reflect current demand, not protocol capacity. Our measurements characterize usage patterns during a specific 90-day window, not protocol performance limits under saturation. Conclusions about maximum throughput should rely on protocol parameter analysis, not observed throughput.

b) Token price volatility: USD-denominated costs are sensitive to the token prices observed during our window. ETH ranged from approximately $2,000 to $4,000, and TIA ranged from approximately $0.30 to $0.85. Different windows would yield different cost figures. We mitigate this by reporting VWAP and percentile distributions alongside snapshot values.

c) Polkadot throughput upper bound: Our Polkadot throughput measurement uses included_candidates × max_pov_size, which is an upper bound since actual PoV sizes are not exposed on-chain. Real throughput is lower; parachains typically use 1–3 MiB per PoV, not the full 10 MiB maximum.

d) Espresso Mainnet 0: Espresso's Mainnet 0 is a permissioned early deployment. Its cost, throughput, and security characteristics should not be compared on equal footing with

permissionless, economically secured networks. We include it for architectural completeness and note its pre-production status throughout.

e) Single observation window: A 90-day window provides a snapshot, not a longitudinal view. Protocol behavior under different market conditions, governance events, or adoption phases may differ. Our open-source tooling enables future researchers to extend the measurement window.

f) Incomplete data for some protocols: Some table entries remain marked TBD where data collection was incomplete at publication time. This reflects the practical difficulty of collecting and normalizing cross-protocol data at scale, a challenge that itself motivates the open-source tooling contribution of this work.

VIII. Conclusion

This paper presented the first comprehensive empirical benchmarking study of six production blockchain DA protocols (Polkadot ELVES, Ethereum EIP-4844/PeerDAS, Celestia, Espresso Tiramisu, NEAR, and Avail) using 90 days of mainnet data collected under a unified methodology.

A. Summary of Findings

Our analysis yields four principal findings.

First, all six protocols operate well below their theoretical capacity ceilings (<50% utilization), indicating that the current DA landscape is demand-constrained. Observed throughput measures ecosystem adoption, not protocol engineering limits. This observation is important for interpreting any cross-protocol throughput comparison: until protocols approach saturation, throughput comparisons reflect market dynamics rather than technical capability.

Second, DA costs span seven orders of magnitude across protocols, from $\sim\$3\times10^{-9}$/MiB (Espresso, whose 1 wei/byte fee is a Mainnet 0 placeholder) to ∼$0.80/MiB (Celestia peak during a demand spike). Annualized cost normalization, which accounts for protocol-specific retention windows, reveals that protocols appearing cheap at spot pricing (due to short retention) may be expensive when continuous availability is required. The absence of a single "cheapest" protocol after annualization underscores the importance of multi-dimensional evaluation.

Third, no protocol implements slashing for pure data withholding. Enforcement is uniformly indirect: Polkadot gates finality on availability bitfields, Celestia employs bad encoding fraud proofs, Ethereum uses peer scoring, Espresso relies on committee reputation, NEAR has an unused challenge mechanism, and Avail slashes only for equivocation. This universal gap reflects the fundamental difficulty of attributing data withholding (an omission fault) to specific actors.

Fourth, DA is universally temporary, with retention windows ranging from ∼85 minutes (Avail) to ∼30 days (Celestia). This ephemerality is a deliberate design choice, but it creates binding constraints for downstream consumers, particularly optimistic rollups whose fraud proof windows may exceed their DA layer's retention period.

B. Architectural Taxonomy

We classified DA architectures along three paradigms (DAS, VID, and validator-set recovery) and along orthogonal dimensions of encoding scheme and commitment type. This taxonomy reveals that the choice of verification paradigm has deeper implications than encoding or commitment choices: it determines light client capabilities, trust assumptions, and enforcement mechanisms. DAS protocols (Celestia, Avail, Ethereum) enable trust-minimized verification but require active network participation. Validator-recovery protocols (Polkadot, NEAR) provide deterministic guarantees but offer no independent light client verification. VID (Espresso) provides algebraic bribery resistance at the cost of committee trust.

C. Contributions and Reproducibility

Beyond the empirical findings, this work contributes: (i) open-source data collection tooling for all six protocols, (ii) raw datasets comprising over 14 million blocks, (iii) analysis notebooks that reproduce all figures and tables, and (iv) interactive dashboards for real-time data exploration. All artifacts are released under Apache 2.0 (code) and CC BY 4.0 (data) licenses to enable reproducibility and longitudinal extension by the research community.

D. Future Work

Several directions emerge from this study. *Longitudinal extension*: as adoption grows and protocols approach saturation, repeating this analysis will reveal supply-constrained dynamics not observable in the current demand-constrained regime. *Stress testing*: synthetic load testing under controlled conditions would complement our observational methodology with causal measurements of protocol limits. *Withholding enforcement*: the universal absence of withholding-specific slashing identifies an open problem; formal analysis of incentive-compatible enforcement mechanisms for omission faults remains an important research direction. *Cross-layer analysis*: measuring end-to-end rollup performance (confirmation time, cost, finality) as a function of DA backend choice would directly inform the rollup developer decision framework. *PeerDAS deployment*: Ethereum's PeerDAS is in progressive deployment; measuring its impact on DA pricing and validator behavior as it reaches full activation is a natural extension of this work.

Acknowledgment

This work was supported by the Web3 Foundation under the W3F Grants Program. The authors thank the Web3 Foundation for their support of open blockchain research. All data, code, and analysis artifacts produced in this study are released under open licenses (Apache 2.0 for software, CC BY 4.0 for data and documentation).

References

[1] M. B. Saif, S. Migliorini, and F. Spoto, "A survey on data availability in layer 2 blockchain rollups: Open challenges and future improvements," *Future Internet*, vol. 16, 2024.

[2] E. Foundation, "Eip-4844: Shard blob transactions," https://github.com/ethereum/EIPs/blob/master/EIPS/eip-4844.md, 2024, accessed: 2026-02-06.

[3] F. D'Amato, D. Feist *et al.*, "Eip-7594: Peerdas – peer data availability sampling," https://github.com/ethereum/EIPs/blob/master/EIPS/eip-7594.md, 2024, accessed: 2026-02-06.

[4] M. Al-Bassam, "Lazyledger: A distributed data availability ledger with client-side smart contracts," arXiv preprint arXiv:1905.09274, 2019, https://arxiv.org/abs/1905.09274.

[5] J. Burdges and A. Cevallos, "Efficient execution auditing for blockchains under byzantine assumptions," https://eprint.iacr.org/2024/961, 2019, https://eprint.iacr.org/2024/961.

[6] E. Systems, "The espresso sequencer: Hotshot consensus and tiramisu data availability," https://docs.espressosys.com/, 2023.

[7] N. Protocol, "The near white paper," https://pages.near.org/papers/the-official-near-white-paper/, 2020, accessed: 2025-09-16.

[8] Avail, "Avail: The data availability blockchain," *Avail Project Documentation*, vol. 1, no. 2.1, 2024.

[9] S. Capital, "A deep dive into data availability: The promises and challenges of scaling web3," https://www.symbolic.capital/writing/a-deep-dive-into-data-availability-the-promises-and-challenges-of-scaling-web3, 2024, accessed: 2025-09-22.

[10] M. Al-Bassam, A. Sonnino, and V. Buterin, "Fraud and data availability proofs: Maximising light client security and scaling blockchains with dishonest majorities," in *arXiv preprint arXiv:1809.09044*, 2019, https://arxiv.org/abs/1809.09044.

[11] A. Kate, G. M. Zaverucha, and I. Goldberg, "Constant-size commitments to polynomials and their applications," in *Advances in Cryptology – ASIACRYPT 2010*, ser. Lecture Notes in Computer Science, vol. 6477. Springer, 2010, pp. 177–194.

[12] J. Burdges, A. Cevallos, P. Czaban, R. Habermeier, S. Hosseini, F. Lama, H. Kilinc, X. Luo, F. Shirazi, A. Stewart, and G. Wood, "Overview of polkadot and its design considerations," 2020.

[13] E. Foundation, "Eip-7691: Blob throughput increase," https://github.com/ethereum/EIPs/blob/master/EIPS/eip-7691.md, 2025, accessed: 2026-02-06.

[14] Avail-Team, "Avail: A unifying blockchain network," *Avail Project Documentation*, vol. 1, no. 1, 2024.

[15] E. Systems, "Bribery-resistant verifiable information dispersal," in *Innovations in Theoretical Computer Science (ITCS)*, 2026, to appear.

[16] V. Buterin and D. Feist, "Danksharding," https://ethereum.org/en/roadmap/danksharding/, 2024, accessed: 2026-02-06.

[17] Polkadot, "Polkadot protocol architecture," https://docs.polkadot.com/polkadot-protocol/architecture/polkadot-chain/overview., 2025, accessed: 2025-09-22.