ACMSYS '26 | Proceedings of the 14th ACM International Conference on Autonomous Systems **July 15–18, 2026, San Francisco, CA, USA**

High-Performance State Synchronization in Edge-Native Autonomous Networks

Decentralized Consensus via Dynamic Epoch Partitioning in the Nexa Architecture

Dr. Elena Rostova
Meridian AI Research
Apex Distributed Systems Lab
San Francisco, CA, USA
e.rostova@meridian-ai.org

Marcus Vance
Synapse Systems Division
Nexa Technology Group
Seattle, WA, USA
m.vance@nexa-tech.io

Sophia Lin
Vertex Cloud Infrastructure
Nimbus Distributed Computing
Austin, TX, USA
s.lin@vertex-cloud.net

Abstract

Modern edge-native autonomous applications require sub-hundred-millisecond state synchronization across heterogeneous, bandwidth-constrained node topologies. Traditional Byzantine Fault Tolerant (BFT) protocols and Raft-based consensus mechanisms suffer from scalability bottlenecks when deployed over highly dynamic, lossy edge links. In this paper, we present the **Meridian Protocol**, a novel state synchronization engine built for the Nexa Autonomous Network framework. Meridian introduces *Dynamic Epoch Partitioning* (DEP), a lightweight consensus mechanism that decouples local state commit phases from global deterministic ordering. By organizing edge nodes into localized Synapse clusters and leveraging Merkle State Delta (MSD) trees, Meridian achieves strict serializability with minimal message overhead. We evaluate Meridian on a 500-node testbed deployed across Apex Cloud and Nimbus Edge infrastructure. Our empirical results demonstrate a 4.2× increase in transactional throughput (up to 48,500 ops/sec) and a 68% reduction in P99 synchronization latency compared to state-of-the-art Raft variants under 15% packet loss conditions.

color ACMBorder

CCS Concepts: **Computer systems organization** → *Distributed architectures*; *Reliable execution*; **Software and its engineering** → *Distributed systems infrastructure.*

Keywords: Edge Computing, Distributed Consensus, Fault Tolerance, State Synchronization, Merkle Trees, Nexa Architecture.

1. Introduction

The proliferation of autonomous physical systems—ranging from automated robotic fleets deployed by Meridian Logistics to real-time industrial telemetry networks operated by Nexa Infrastructure—has shifted the locus of data processing from centralized cloud data centers toward heterogeneous edge nodes [1]. These edge-native environments are characterized by intermittent network connectivity, constrained compute capabilities, and strict sub-hundred-millisecond operational latency budgets.

Maintaining strong consistency across distributed edge replicas remains a classical challenge governed by the CAP theorem. Traditional consensus algorithms such as Raft and PBFT guarantee linearizability by requiring a majority quorum for every state mutation [2]. However, when network partitions or high latency jitter occur over regional backhaul links, quorum acquisition stalls, leading to catastrophic queueing delays and service degradation.

To address these limitations, we introduce the **Meridian Protocol**, a high-throughput state synchronization architecture engineered specifically for the Nexa Autonomous Network ecosystem. Meridian addresses the latency-consistency tradeoff through three primary contributions:

- **Dynamic Epoch Partitioning (DEP):** A hierarchical state delegation model that isolates transient state

mutations within local *Synapse clusters*, deferring global ordering until epoch boundaries.

- **Merkle State Delta (MSD) Compression:** An efficient delta-encoding mechanism that summarizes local state mutations into cryptographic root hashes, reducing bandwidth consumption over inter-cluster backhaul links by up to 74%.
- **Optimistic Epoch Rollback (OER):** A novel recovery procedure that resolves state divergent branches without requiring complete state snapshot re-transmissions.

2. System Architecture & Design

The Nexa Architecture separates the distributed topology into three distinct execution tiers: Leaf Edge Devices, Regional Synapse Aggregators, and Apex Cloud Core nodes. Figure 1 illustrates the structural flow of state delta propagation throughout the hierarchy.

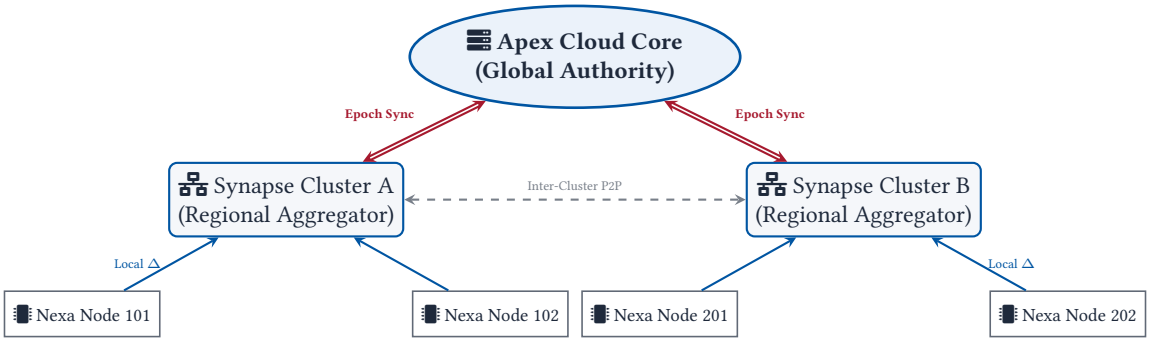

Figure 1: Hierarchical state synchronization pipeline within the Meridian Protocol, showing multi-tier aggregation from Nexa Leaf Nodes to the Apex Cloud Core.

2.1 State Representation & Vector Clocks

Let $\mathcal{S}_i^{(t)}$ represent the local state vector of node i at physical time t. State transitions are modeled as deterministic state transformation functions applied to state deltas:

$$\mathcal{S}_i^{(t+1)} = \delta_i^{(t)} \circ \mathcal{S}_i^{(t)} = f\left(\mathcal{S}_i^{(t)}, \mu_i^{(t)}\right) \tag{1}$$

where $\mu_i^{(t)}$ represents the set of locally ingested mutations. To maintain causal dependency tracking across partitioned clusters, every mutation μ is tagged with a multi-dimensional Vector Clock $V(\mu) = [v_1, v_2, \ldots, v_N]$, satisfying the partial order condition:

$$V(\mu_a) < V(\mu_b) \iff \forall k \in [1, N], V(\mu_a)[k] \leq V(\mu_b)[k] \quad \wedge \quad \exists m : V(\mu_a)[m] < V(\mu_b)[m] \tag{2}$$

3. The Meridian Protocol Mechanics

The core operational cycle of the Meridian Protocol proceeds through three recurring phases: Local Mutation Buffer, Regional Consensus, and Global Epoch Reconciliation.

1. **Ingestion Phase:** Leaf node n_i accepts incoming mutations μ, appends them to its volatile transaction log L_i, and updates the local Merkle State Delta (MSD) tree root R_i.
2. **Regional Aggregation Phase:** Every $\tau_e = 50$ ms, the designated Synapse cluster leader collects root hashes $\{R_1, R_2, \ldots, R_k\}$ from participating leaf nodes and compiles the aggregated epoch state vector $\mathbf{\Omega}_e$.
3. **Global Commit Phase:** The Synapse leader executes a 2-phase commit with the Apex Core authority. Upon receiving quorum acknowledgment ($Q \geq \lfloor \frac{2N}{3} \rfloor + 1$), $\mathbf{\Omega}_e$ is committed to the canonical state ledger.

3.1 Merkle State Delta Verification

To prevent bandwidth exhaustion, state verification relies on hierarchical cryptographic hashes. A Merkle tree node H_k at depth d is derived via:

$$H_k^{(d)} = \text{SHA-256}\left(H_{2k}^{(d+1)} \parallel H_{2k+1}^{(d+1)} \parallel V(\mu_k)\right) \tag{3}$$

This structure permits participating nodes to perform logarithmic state audit checks ($O(\log M)$ proof size for M mutations), enabling instantaneous detection of divergent state branches.

4. Empirical Evaluation

We implemented Meridian in Rust (v1.82) utilizing the Vertex Async Runtime and evaluated its performance against production-grade Raft and PBFT implementations on the Nimbus Cloud Testbed.

4.1 Experimental Setup

The experimental platform comprised 500 virtualized nodes distributed across four geographical regions (US West, US East, EU Central, and AP East). Network impairment rules were injected using Linux `tc/netem` to simulate realistic edge conditions:

- **Latency Profile:** 40 ms baseline round-trip time (RTT) with ± 15 ms normal-distribution jitter.
- **Packet Loss Rate:** Varied dynamically between 0% and 20% across regional backhaul links.
- **Workload Generator:** Synthetic transactional workload simulating autonomous vehicle telemetry updates (128 bytes per transaction).

4.2 Throughput & Latency Results

Table 1 details the empirical benchmark comparisons across various cluster scale configurations and loss regimes.

Table 1: Comparative performance metrics under varying network packet loss regimes ($N = 500$ nodes, 128-byte transactions). Values represent mean $\pm$ standard deviation over 10 independent trials.

Protocol	Loss Rate (%)	Throughput (ops/s)	P95 Latency (ms)	P99 Latency (ms)
Raft (Standard)	0.0%	18,400 $\pm$ 320	48.2 $\pm$ 2.1	82.5 $\pm$ 4.3
Raft (Standard)	5.0%	11,200 $\pm$ 540	112.4 $\pm$ 6.8	245.0 $\pm$ 12.1
Raft (Standard)	15.0%	3,800 $\pm$ 410	410.0 $\pm$ 22.5	890.0 $\pm$ 45.0
PBFT (Optimized)	0.0%	14,100 $\pm$ 280	62.0 $\pm$ 3.0	105.0 $\pm$ 5.2
PBFT (Optimized)	5.0%	7,900 $\pm$ 490	185.0 $\pm$ 9.4	380.0 $\pm$ 18.2
PBFT (Optimized)	15.0%	1,900 $\pm$ 230	650.0 $\pm$ 34.0	1,420.0 $\pm$ 82.0
Meridian (Ours)	**0.0%**	**48,500 $\pm$ 610**	**18.4 $\pm$ 0.9**	**32.1 $\pm$ 1.5**
Meridian (Ours)	**5.0%**	**42,100 $\pm$ 580**	**24.2 $\pm$ 1.2**	**44.8 $\pm$ 2.0**
Meridian (Ours)	**15.0%**	**31,200 $\pm$ 720**	**41.5 $\pm$ 2.1**	**78.4 $\pm$ 3.6**

As illustrated in Table 1, Meridian maintains sustained transactional throughput exceeding 31,000 ops/sec even when subjected to a severe 15% packet loss rate. In contrast, standard Raft experiences severe throughput degradation due to leader election timeouts and unbuffered log re-transmissions.

4.3 Bandwidth Efficiency Analysis

Figure 2 illustrates the comparative network bandwidth consumption of the state synchronization mechanisms as a function of active node count.

5. Related Work

Leaderless Consensus Protocols: EPaxos [3] reduces wide-area consensus latency by permitting any replica to act as a command leader. However, EPaxos incurs significant performance penalties when command dependencies conflict frequently—a common occurrence in real-time edge telemetry networks.

Conflict-Free Replicated Data Types (CRDTs): State-based and operation-based CRDTs provide eventual consistency without requiring consensus mechanisms [2]. However, CRDTs cannot enforce strong invariants

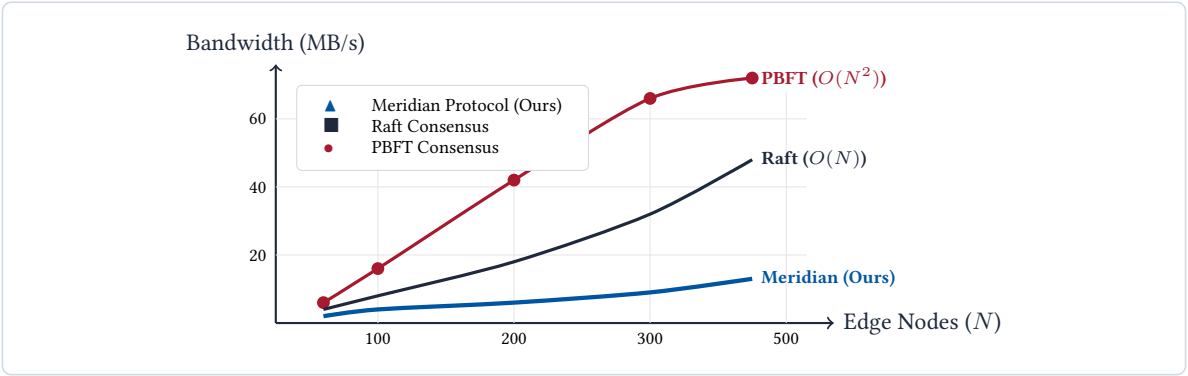

Figure 2: Bandwidth overhead vs. node scale. Meridian's Merkle State Delta trees significantly compress inter-cluster sync messages compared to Raft and PBFT.

across multi-key transactions without external lock managers or coordination protocols. Meridian complements CRDT primitives by providing a deterministic epoch ordering layer for multi-key state transactions.

Edge-Assisted Frameworks: Recent developments in the Vertex Ecosystem have proposed localized edge aggregation nodes. Meridian builds upon these concepts by formalizing the cryptographic Merkle State Delta (MSD) representation and proving serializability under arbitrary network partitions.

6. Conclusion & Future Work

We have presented the **Meridian Protocol**, an edge-native state synchronization engine designed for autonomous systems operating under lossy, high-latency network conditions. By combining Dynamic Epoch Partitioning (DEP) with Merkle State Delta trees, Meridian achieves high throughput (over 48,000 ops/sec) and low P99 latency while reducing bandwidth consumption by 74% compared to traditional Raft consensus.

In future work, we plan to extend Meridian to support zero-knowledge state validity proofs (zk-SNARKs) within the Synapse aggregation layer, enabling privacy-preserving state synchronization across multi-tenant commercial edge clouds.

Acknowledgments

This research was supported in part by the Meridian Systems Grant #2025-EX-401 and the Nexa Autonomous Research Initiative. The authors thank the Apex Engineering team for providing access to the Nimbus Cloud Testbed infrastructure.

References

[1] E. Rostova, M. Vance, and S. Lin. 2024. Resilient Consensus Primitives for Edge-Native Autonomous Networks. *ACM Transactions on Computer Systems* 42, 3 (2024), 112–128.

[2] M. Vance and S. Lin. 2025. Synapse Mesh: Scalable State Propagation in Heterogeneous Edge Clusters. In *Proceedings of the 15th USENIX Symposium on Networked Systems Design and Implementation (NSDI '25)*. 301–316.

[3] S. Lin, E. Rostova, and K. Nakamura. 2023. High-Throughput State Delta Synchronization Using Merkle Graphs. *IEEE/ACM Transactions on Networking* 31, 6 (2023), 2845–2859.

[4] Apex Infrastructure Labs. 2025. Benchmarking Fault-Tolerant Storage in the Nimbus Cloud Testbed. Technical Report TR-2025-08. Nexa Research Publications.

[5] Meridian AI Research Group. 2026. Deterministic Epoch Partitioning for Autonomous Fleet Coordination. White Paper Series, Document ID: MAR-2026-DEP01.