PATENT APPLICATION

INVENTOR(S)

[0001] KIMZEY, Samuel C

TITLE

[0002] BlockMesh Decentralized Memory-Mesh Architecture

TECHNICAL FIELD

[0003] This invention relates to multi-chain blockchain architectures and distributed ledger systems, and more particularly to methods for anchoring and querying a decentralized mesh of memory logs.

BACKGROUND

[0004] Conventional blockchains record all activity in a single chain, leading to scalability bottlenecks and single points of failure. As applications diversify, there is a need for an architecture that maintains parallel memory logs per entity while ensuring global integrity and enabling efficient cross-chain verification.

SUMMARY

[0005] The BlockMesh Decentralized Memory-Mesh Architecture comprises:
[0006] Establishing, per entity, a personal subchain storing sequential event logs.
[0007] Periodically committing each subchain's state root or anchor hash to a base layer blockchain via on-chain mint transactions.
[0008] Forming a cryptographic mesh by linking subchain anchors into the base layer.
[0009] Enabling cross-mesh queries that retrieve and verify subchain logs using cryptographic proofs against the base anchors.

DETAILED DESCRIPTION

[00010] At the conceptual level, each participant entity maintains its own memory subchain recording events. Each subchain periodically emits an anchor—such as a chained hash, pointer, or optional Merkle root—that is committed to a common base layer. These anchors collectively form a mesh, ensuring data integrity and non-repudiation across chains. Cross-mesh queries assemble cryptographic proofs from one or more subchains and verify them against the base layer anchors, allowing participants to efficiently retrieve and trust historical records.

METHOD

[00011] Step 1: Subchain Instantiation – Initialize a personal subchain for each entity to record sequential event logs.
[00012] Step 2: Anchoring – At regular intervals or based on memory-flow triggers, commit each subchain's state root or anchor hash to the base layer using a one-action, one-mint transaction.
[00013] Step 3: Mesh Formation – Maintain cryptographic hash links between subchain anchors and base layer transactions to establish a cohesive mesh.
[00014] Step 4: Cross-Mesh Query – Upon request, collect cryptographic proofs (e.g., anchor paths) from relevant subchains and verify against corresponding base layer anchors to authenticate events.

NARRATIVE WORKED EXAMPLE

[00015] Imagine three organizations—OrgA, OrgB, and OrgC—each operating its own subchain. At t1, OrgA anchors hash H1; at t2, OrgB anchors H2; at t3, OrgC anchors H3. A user seeks proof of an event recorded by OrgB. The user retrieves OrgB's proof path: Event record → block hash → subchain anchor H2, then verifies H2 was committed to the base layer at t2. The mesh ensures the user can trust OrgB's event via the collective anchors.

ALGORITHMIC WORKED EXAMPLE

[00016] In an optional embodiment, anchors are Merkle roots. For a subchain with block data blocks B1–B4, compute Merkle root M = MerkleRoot(B1||B2||B3||B4). Embed M in a base layer mint transaction. To prove inclusion of B3, provide a Merkle proof [Hash(B3), sibling hashes], which verifies against M stored on-chain.

EMBODIMENTS

[00017] Anchoring using chained hashes or pointers without Merkle structures.

[00018] Dynamic anchoring intervals driven by memory-flow pressure metrics.

[00019] Identity-anchored subchains using biometric hashes or external oracle signatures.

[00020] Hierarchical multi-layer subchains (L2–L6) anchoring to intermediate layers before base layer.

IMPLEMENTATION NOTES

[00021] Each anchor commit uses a one-action, one-mint on-chain transaction. No off-chain storage or endpoints are required. All subchain logs and anchors persist permanently, creating an immutable memory mesh. Modules read and verify anchors in-chain.

[00022] Imagine a layered architecture with six chains (L1–L6) anchored by a common Genesis Block at Layer 0.

[00023] At t1, L1 emits anchor A1 to the Genesis Block; at t2, L2 emits anchor A2 linking to A1; continuing up to L6 emitting A6 in sequence.

[00024] An event on L3 at t3 creates a cryptographic attestation attached to A3, verifiable through the chain of anchors back to Genesis.

[00025] Similarly, an event on L5 at t4 attaches to A5, interwoven via L4 and L3 anchors into the mesh.

[00026] Observers tracing proofs from L3 or L5 reconstruct the layered anchor path: Event → A3 → A2 → A1 → Genesis, demonstrating integrity across layers.

[00027] Peaks in cross-layer event density—such as simultaneous events on L2, L4, and L6—naturally form pressure points at the mesh's core.

CLAIMS

1. A method for managing decentralized memory across a plurality of subchains, the method comprising:
    a. establishing, by a processor, a plurality of personal subchains, each subchain storing sequential event logs for a respective entity;
    b. periodically committing, by the processor, a state root or anchor hash of each personal subchain to a common base layer blockchain via a single mint transaction;
    c. forming, by the processor, a mesh of anchored subchains via cryptographic hash links between subchain anchors and base layer transactions;
    d. responding, by the processor, to cross-mesh query requests by retrieving cryptographic proofs from one or more personal subchains and verifying the proofs against corresponding base layer anchors.
2. The method of claim 1, wherein committing a state root or anchor hash comprises generating an anchor hash from subchain block data and including the anchor hash in a mint transaction.
3. The method of claim 1, further comprising associating each personal subchain with an identity anchor selected from a biometric hash or an external oracle signature.
4. The method of claim 1, further comprising dynamically adjusting the commitment intervals for each personal subchain based on aggregated memory-flow metrics.
5. The method of claim 1, wherein the mesh comprises multiple hierarchical layers, and subchains of higher layers anchor to subchains of lower layers to form a layered anchoring structure.
6. The method of claim 1, wherein the anchor hash comprises a Merkle root of the subchain block data.
7. The method of claim 1, further comprising querying Layer 3 trait subchains across multiple personal chains to infer mesh-wide pressure conditions.

ABSTRACT

[00028] A foundational framework for decentralized memory management using a mesh of personal subchains anchored into a base layer, where each subchain's state anchors forge an emergent, verifiable memory substrate without reliance on a single monolithic chain.