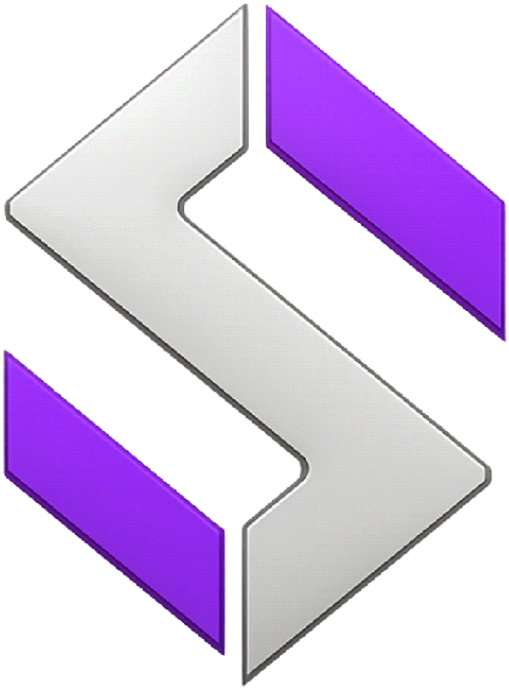


Sentinent

Build Smarter Systems with Decentralized Structured Storage

1. What is Sentinent?

Sentinent is a next-generation, fully decentralized knowledge infrastructure purpose-built for agents, LLM-powered tools, and AI-native workflows. More than mere storage, Sentinent fuses decentralized file persistence, semantic context, cryptographic privacy, and programmable permissions into a composable protocol layer. Every uploaded data object is transformed into an intelligence-ready artifact—structured, machine-readable, owner-controlled, and auditable by design.

Leveraging the **Model Context Protocol (MCP)**, Sentinent automates the annotation, provenance tracking, and cryptographically secure indexing of every file. This approach not only enables structured and dynamic data retrieval, but sets the foundation for autonomous and federated machine reasoning, cross-agent memory, and context-aware LLM orchestration—far surpassing flat file storage or custodial cloud models.

Sentinent's architecture is designed for high-resilience, vendor-neutrality, and "privacy without trade-offs." Users interact via wallet-based identities; every read, write, permission grant, or access revocation is guaranteed by on-chain programmable contracts and zero-knowledge cryptography. Data undergoes local encryption before even entering the peer-to-peer network, and composable APIs let developers and agents plug in advanced workflows without risking centralized lock-in or loss of data sovereignty.

2. Vision and Value Proposition

Sentinent's mission is to build a web-scale, privacy-first, intelligence-aware data layer—the foundation of a decentralized AI future. In this paradigm, 'data' is not just payload, but living, permissioned knowledge, ready for RAG, model retraining, federated analytics, or rapid, autonomous orchestration.

- **Autonomous Contextual Intelligence:**
  Sentinent eliminates the "flat blob" problem by providing semantic metadata, versioning, and search hooks at the protocol level. LLMs and agents can now access, query, and reason directly over files, models, and multi-step histories without patchwork ETL or manual pipelines.
- **End-to-End Sovereignty:**
  Using strictly wallet-based authentication and on-chain policies, Sentinent hands users, teams, and agents direct control over *who accesses what* and *when*—down to individual files, time windows, or action scopes. All delegation and permission models (single-user,

multi-signature, DAO, temporary agent delegation) are fully programmable and recorded for audit and recovery.

- **Permanence Meets Privacy:**
  Cold storage pools and archiving use economic incentives (staking and payment channels) for low-cost, high-durability persistence, while zero-knowledge encryption and dynamic permission trees prevent unauthorized reads—by anyone, including storage nodes, network validators, or third-party actors.
- **Composability Above All:**
  Every API is open, deterministic, and candidate for async or real-time orchestration—allowing on-chain agents, cross-dApp automation, semantic search overlays, and custom dashboarding. Plug Sentinent into bot frameworks, DAO tools, compliance backends, or sovereign research cloud—no silos or vendor risk.

3. Technical Architecture & Protocol Layers

3.1. Model Context Protocol (MCP)

Core Functions:

- **Semantic Tagging:** Incoming files parsed by LLM-powered or plugin-based pipelines; context tags (timestamp, source, topic, entity mentions, signatures) embedded and cryptographically signed.
- **Provenance Graphing:** Each data object is assigned a root hash (Merkle-proofed), with parent/child links for versioning, citation, workflow step mapping, or collaborative audit trails.
- **Composable Metadata Schema:** MCP supports arbitrary extension—teams can define custom semantics (research domain, compliance tags, workflow identity) as JSON-LD or linked data, usable by future dApps.

Technical Detail:

- All metadata is stored as encrypted subfields within the primary file envelope (e.g. LibP2P blocks, IPLD or equivalent).
- MCP-compliant data is queryable natively via LLM plugins, agent interpreters, or on-chain indexers (e.g. EIP-712-compatible schemas for cross-chain retrieval).
- Semantic context can be generated at ingest (client-side embedding, auto-tag) or appended post-hoc via permissioned agent actions.

3.2. Zero-Knowledge Encryption Layer

File Encryption:

- Files are chunked, encrypted locally using wallet-controlled keys (ChaCha20, AES-256-GCM), and then distributed to P2P storage pools (e.g. IPFS, Filecoin, Arweave, extensible with hypersync/other backend).
- **Permission Management:**
  Each access grant is programmatically expressed as a *capability*

invocation—permissioning rules (who can decrypt, for how long, at what cost) are signed and posted on-chain. *Revocation* is possible at key-rotation, via time-locked or event-based logic.

Zero-Knowledge Claims:

- For sensitive data/AI use-cases (medical datasets, compliance), retrieval and computation can be wrapped in zk-SNARK proofs. This allows model training, retrieval-augmented generation (RAG), or even aggregate analytics **without exposing plaintext to storage nodes or computation providers**.
- All claims, access events, and cryptographic proofs are time-stamped and optionally published to a public, tamper-evident audit log.

3.3. Decentralized Storage & Staking Market

Storage Participation:

- Storage providers stake Sentinent tokens and signal available resources (disk, bandwidth, uptime SLA).
- Data owners broadcast storage orders (with redundancy/archival duration parameters); economic matching is achieved via open P2P auctions or fixed-price market pools.
- **Slashing/Rewards:**
  Nodes must regularly produce cryptographic proofs of storage (e.g. PoRep, PoSt) and file integrity (Merkle challenge or LLM challenge). Failure (lost data, downtime) results in slashing; reliability is rewarded each epoch.

Cold Storage Pools:

- For large datasets/model backups, storage is sharded/redundant and retrievable via short, programmable proofs (SPoR, similarity search, or agent-driven consensus).
- Multi-chain payment bridges allow users to pay fees in various assets (e.g. native tokens, stablecoins).

3.4. Wallet-Based Access & Role Management

- **Authentication:**
  All access is mediated through cryptographic wallets (EVM-compatible, Solana, MultiChain connectors). No usernames/passwords, no admin resets.
- **Delegation:**
  Agents, teams, bots, or DAOs can be authorized per-file, per-session, or via time-bound policies.
- **Group Permissions & Multi-Sig:**
  Advanced workflows allow multi-key access (e.g. team knowledge vaults, corporation data pods, agent mesh collaborations).

4. Workflow Examples & Use-case Deep Dive

4.1. AI Agent Memory Planting

- **Upload:**
  Agent writes summary/decision history to wallet, tags with context ("strategy update, July 2025, market: Solana").
- **Storage:**
  File is encrypted, archived in cold storage pool, and semantic index is updated.
- **Retrieval:**
  Months/years later, agent or supervisor LLM queries "last 10 policy updates X context"—instantly retrieves semantic objects, summarizes, reasons, or triggers alert.

4.2. Collaborative Research / DAO Knowledge Base

- **Collaborator uploads research draft, tags as "proposal, WIP, governance 2025".**
- **Multi-sig access:**
  DAO core team can review, annotate, or vote—all access/review events are logged, encrypted, and only available to permitted signers.
- **Publication:**
  Once consensus reached, knowledge object is permissioned for public DAO access or RAG-enabled inference (retrospective, onboarding bench, etc.).

4.3. RAG Model Data Pipeline

- **Historical Q&A corpus uploaded, tagged by domain, date, role, trust score.**
- **Model pipeline access:**
  During prompt completion, LLM or RAG pipeline queries Sentinent index for "all verified, high-trust answers on topic X since 2023".
- **Private Retrieval:**
  Only wallet acting as model oracle or explicitly-permissioned agent receives dataset; no third party (including stakers) ever sees plaintext.

5. Advanced Protocol Considerations

- **Multi-Chain Settlement:**
  Sentinent can handle fee settlement, staking, and incentive claims cross-chain. For example, a model provider on Solana can pay for archiving on Filecoin via asset bridges, with proofs posted to both ledgers.
- **Programmable Permissions (PBAC):**
  All file permissions are Turing-complete: expand to workflow gates (e.g. “unlock X if Y agent succeeds” / “revoke on DAO vote outcome”) or adapt to enterprise RBAC integration.
- **Dynamic Semantic Indexing:**
  Developers or DAOs can add real-time or retrospective MCP tags—enabling ongoing annotation, workflow automation, compliance updates without re-uploading or breaking file provenance.
- **On-Chain Auditing and Forensics:**
  All data accesses, permission events, and protocol-level actions generate immutable,

queryable log objects—crucial for enterprise attestation, legal compliance, research reproducibility, or forensic analysis.

6. Tokenomics & Economic Security

- **Token Utility:**
  $SENT token is core for payment (storage fees, indexing, search), staking, slashing, and decentralized protocol governance (treasury, upgrade votes, bug bounty allocation).
- **Incentive Alignment:**
  Honest, performant nodes accrue rewards by proving service and availability. Slashing deters freeloading, data loss, or malicious bridging.
- **Governance Path:**
  Roadmap changes, fee parameters, protocol upgrades pass via on-chain proposals, weighted by stake, incentivizing long-term community participation and preventing protocol capture.

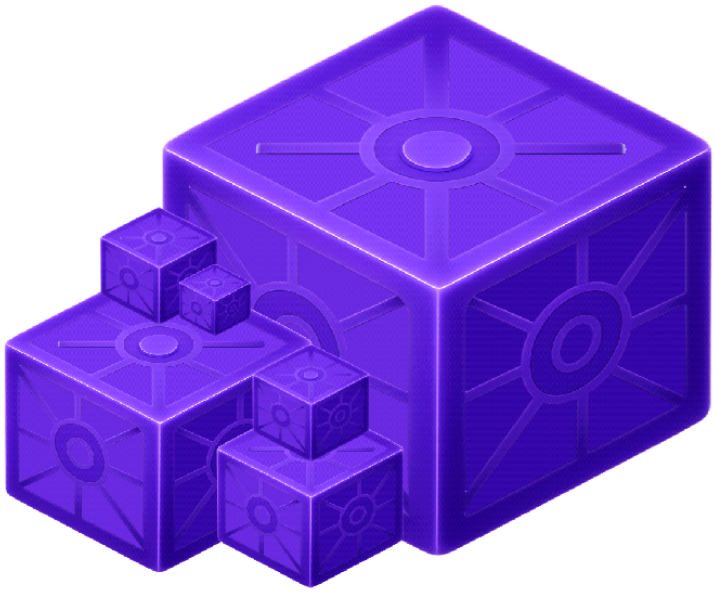
7. Roadmap

- **Phase 1: Foundations of Intelligent Storage:**
  - Launch Sentinent v1 with cold storage, semantic file layer, and wallet access
  - Enable zero-knowledge encryption by default for all uploads
  - Introduce MCP v1 (Model Context Protocol) for file enrichment
  - Begin onboarding early AI builders and agent projects
- **Phase 2: AI Integration & Contextual Retrieval**
  - Deploy RAG-ready file layer with context-aware retrieval support
  - Launch agent memory interface
  - Integrate cold archival tools for AI workloads and historical data
  - Improve MCP tagging accuracy and semantic auto-indexing
- **Phase 3: Collaboration and Context-Aware Access**
  - Add query logging and usage analytics for AI systems
  - Begin protocol-level performance optimization for large semantic datasets
  - Launch feedback program with selected AI research partners
  - Roll out staking mechanism to secure the decentralized infrastructure
- **Phase 4: Composability, Discoverability & Scale**
  - Integrate MCP v2 with higher-level ontologies and contextual search
  - Expand network resilience and node decentralization
  - Execute integration with select LLM providers for native RAG pipelines
  - Release Sentinent Explorer to audit structured data state across the network

8. References & Further Reading

- Sentinent Protocol Docs
- Model Context Protocol (MCP) technical standard (see documentation)
- Zero-Knowledge cryptography sources: ZK-SNARKs, Halo2, PlonK
- Filecoin/IPFS persistent storage proofs and PoRep, PoSt specs
- Wallet Auth models for access delegation
- Decentralized agent/ecosystem integration: RAG, LLM plugin, agent mesh

Ready to get involved?



Explore the detailed developer docs, join the protocol forums, or contribute plugins/extensions—Sentinent is open-source, auditable, and extensible for every use-case in the decentralized, AI-native world.

If you require this breakdown as individual docs/chapters for wiki or Docusaurus format, just specify—full technical reference and schema samples available on request.