TECHNICAL & ECONOMIC WHITEPAPER

The Inaya Protocol

A decentralized sovereign custody network for high-value enterprise and personal data assets.

Document INAYA-WP-2026-V2 · Classification Public · August 2026

SECTION 01

Abstract

Contemporary high-value enterprise data storage carries a structural single-point-of-failure risk: centralized cloud platforms hold complete, whole files, meaning a single breach of infrastructure or credentials exposes the entire asset.

The Inaya Protocol eliminates this by never letting a complete, readable copy of a file exist anywhere outside the owner's own device. Every file is encrypted locally, split into two independent shards, and pinned to decentralized storage — with only a tamper-evident ownership record, never the file itself, ever touching the blockchain.

This document covers the cryptographic mechanics, the on-chain custody model, the token economics, and — new since the protocol's original whitepaper — the application layer now built on top of it: a Business SaaS workspace, a decentralized security/threat-intelligence layer, an educational platform, an investor data room, two desktop apps, and three purpose-built AI assistants. All of it runs on BNB Chain Testnet today; nothing in this document describes a mainnet-live system yet.

SHARD VECTORS PER ASSET	ENCRYPTION	$INAYA TOTAL SUPPLY
2	256-bit AES-GCM	30,000,000 (fixed)

SECTION 02

The Problem With Centralized Storage

When a file is saved to a classical server, it resides whole. If the administrative boundary is breached, the cleartext contents are entirely compromised.

PERIMETER EXFILTRATION

Attackers targeting network interfaces to capture whole files in transit.

MULTI-TENANT CONTAMINATION

Hardware-level bleeds inside shared hypervisors, exposing one customer's data to another.

CUSTODY KEY FORGERY

Forced access to a central database holding system-managed master encryption keys.

> "Inaya" is derived from the Arabic term for absolute protection, vigilant care, and functional grace — the operating philosophy behind treating storage as an active guardian process rather than a passive database write.

SECTION 03

Client-Side Encryption & Sharding

Files never leave the user's device in a unified, readable form. Every step below happens locally — verified directly against the custody-sdk source, not assumed.

KEY DERIVATION

```
Kv = PBKDF2(passkey, salt, iterations = 100,000, HMAC-SHA256, len = 256)
```

The master passkey (P) and a fresh, cryptographically random 16-byte salt (S) derive a 256-bit AES key locally — the passkey itself is never transmitted anywhere, at any point, in either direction.

The file is then encrypted with AES-GCM-256 and the resulting ciphertext (C) is bisected at its exact midpoint into two shards:

BINARY SHARDING

```
Mp = floor(length(C) / 2)
Shard Alpha = C[0 ... Mp-1]
Shard Beta  = C[Mp ... end]
```

> Neither shard alone contains a contiguous, decryptable structure — an isolated compromise of a single storage node yields random binary noise, not a usable file fragment. This is literal midpoint bisection, not erasure coding.

SECTION 04

On-Chain Ledger Attestation

Inaya eliminates the need for a central database cluster for asset ownership — the registry lives on-chain, tracking only shard pointers (IPFS CIDs) and a file hash, never file content.

REGISTRY WRITE (INAYACUSTODY)

```
function batchRegisterAssets(
  bytes32[] fileHashes,
  uint256[] fileSizes,
  string[] shardACIDs,
  string[] shardBCIDs
) external
```

> This write is permanent by design — there is no update or delete function for a registered asset. Renaming, moving, or deleting a file is an off-chain metadata operation, not a contract mutation.

SECTION 05

Pay-As-You-Go Storage

- Storage is billed in stablecoin (USDT) to remove multi-token friction for developers and retail users.
- Fee rates are read live from the deployed contract (usdtFeePerGB / inayaFeePerGB) rather than hardcoded — the protocol can adjust pricing over time without a client update.
- Current live rate, read directly from the deployed contract: 0.0044 USDT/GB (~4.50 USDT/TB) for storage ingress. $INAYA carries zero ingress fee today; $INAYA egress/retrieval fees activate at mainnet, not before — confirmed directly by the founding team, not a testnet limitation.
- Staking $INAYA unlocks priority bandwidth routing and a share of network fee yield (Section 09).

SECTION 06

Corporate Reserve — Annual Plans

For institutions with fixed large-scale storage needs. This is the real, implemented tier structure — three fixed annual allocations, not an API-call-volume tier system.

ALLOCATION	RESERVE FEE (USDT/YR)	ANNUAL MAINTENANCE (INAYA-EQ/YR)
250 TB / Year	13,500	500
500 TB / Year	27,000	1,000
1000 TB / Year	54,000	2,000

> These are the real, live tier figures shown in the dApp's Business Model tab and checkout flow as of this writing.

SECTION 07

Revenue Distribution — RevenueRouter & Corporate Escrow

Corrected from the prior edition of this document, against the real transaction flow in the dApp's checkout code.

1. **1.** A corporate customer approves USDT and calls RevenueRouter.processCorporateInvoice(usdtAmount) — this is a real, standalone on-chain transaction.
2. **2.** The client then separately approves USDT and calls InayaCorporateEscrow.createEscrow(corporate, node, cogsAmount) for the node-operator's COGS share — a second, independent transaction, not an automatic cascade triggered by the router itself.
3. **3.** InayaCorporateEscrow locks that allocation and releases it over 12 fixed monthly payouts (releaseMonthlyPayout(), callable by anyone once a release is due).

> **Correction, stated plainly.** An earlier version of this material described the router→escrow handoff as happening "atomically within the same blockchain transaction" with RevenueRouter "automatically" calling createEscrow(). That is not what the deployed contracts do — it's two separate transactions, both client-initiated. The revenue-split percentages themselves (39% node / 51% treasury / 10% team) are not a marketing estimate: the 39% node-operator share is hardcoded in production settlement code (src/app/api/stripe-webhook/route.js: cogsAmountWei = (invoiceAmountWei * 39n) / 100n), and the remaining 61% splits 51%/10% between Treasury and Team, reconciling exactly with the protocol's own published EBITDA math (61% gross margin − 3% grants − 5% R&D − 2% admin/legal/marketing = 51%).

SECTION 08

Node Performance & Proof of Storage

- InayaNodeRegistry tracks each node's capacity, tier, uptime, and stake — but per the contract's own header comment, this is coordinator-verified telemetry (an authorized verifier wallet attests it), not cryptographic proof-of-storage. Described accurately here, not oversold.
- InayaProofRegistry separately stores a Merkle root per asset and verifies chunk-level proofs — today this verification is backend-checked (onlyOwner); the contract's own comment documents a planned path to make it permissionless and stake-slashing-backed, not yet built.
- Settlement is deliberately timelocked: a verifier queues a settlement (computing commission, not moving funds), and only after a 36-hour delay can anyone call releaseSettlement — protecting the reserve from a single compromised key.

SECTION 09

Token Economics

ALLOCATION	%	TOKENS
Swarm Reserve (Node Rewards)	40.0%	12,000,000 INAYA
Staking Rewards Pool	26.7%	8,000,000 INAYA
Liquidity Pool	21.7%	6,500,000 INAYA
Team Runway	5.0%	1,500,000 INAYA
Core Ecosystem Fund	3.3%	1,000,000 INAYA
Genesis Airdrop	3.3%	1,000,000 INAYA

> Fixed 30,000,000 total supply, consistently published across every prior team document and cross-checked here: the six allocations above sum to exactly 30,000,000 tokens.

Swarm Reserve emissions are uptime-gated, not automatic:

- A 3-month, 90%-uptime commitment cliff before a node qualifies for any Swarm Reserve emission.
- Monthly cap by tier: 30 $INAYA (98%+ uptime), 20 $INAYA (95–97.9%), 10 $INAYA (90–94.9%), 0 below 90%.
- InayaNodeRegistry's tier/commission logic is confirmed directly in the deployed contract code; the emission-cap schedule above is the tokenomics design that logic implements.

SECTION 10

Beyond The Protocol — The Application Layer

Everything above is the foundation. Since the protocol's original whitepaper, a full application layer has shipped on top of it — most of which a user never has to know is running on a blockchain at all.

- Business Workspace — a B2B SaaS product: organizations, departments, projects, and documents with server-enforced approval workflows, granular permissions, secure sharing, and a permission-aware AI assistant. Email sign-in, no wallet required.
- Business Operations & Finance/HR — Tasks, CRM, Procurement, and Inventory on the same permission foundation, plus a Finance & HR layer (invoices, expenses, payments, employee records, leave management) — a testnet demonstration layer today, not regulated banking or payroll.
- Security Layer ("Inaya Firewall") — a decentralized, node-reported threat-intelligence network with reputation-weighted confirmation and on-chain-anchored verdicts, surfaced on a public web page, in the mobile app, and via real OS-level enforcement on desktop.
- Inaya Learn — a YouTube-based educational discovery platform with an AI tutor, built to make the apps a daily-use destination beyond storage and staking.
- Investor Data Room — a branded, access-controlled document room with per-visitor engagement tracking.
- Two Tauri desktop apps — thin native wrappers around the Business Workspace and the main dApp, with system tray, native notifications, and signed auto-updates.
- Four Gemini-powered AI assistants — Docs, Business, Security, and Learn — sharing one technical pattern but different guardrail philosophies suited to their purpose; Docs, Security, and Learn are grounded by a shared RAG retrieval layer (see the Ecosystem Architecture document for full detail).
- Oracle & Automation Layer — an on-chain registry of approved data sources plus a self-operating keeper that executes pre-approved contract actions under smart-contract rules, not manual admin commands. Deployed and running live on BSC Testnet, publicly verifiable at inayanetwork.com/automation.

> Full technical depth on every one of these lives in the companion "Complete Ecosystem Architecture" document — this whitepaper stays focused on the core protocol.

SECTION 11

Leadership

Talha Waqas — Founder & CTO

Web3 full-stack engineer specializing in client-side cryptographic infrastructure, EVM smart contract architecture, and secure data-routing pipelines. Drives core protocol development end to end.

Yakub Adnan — Co-Founder & Growth Lead

Web3 growth operator and community strategist specializing in DePIN, user acquisition, and AI-driven ecosystem scaling. Leads growth architecture, community operations, and campaign distribution.

Fibha Urooj — CFO

Leads financial planning, budgeting, compliance, and operational finance — building the financial foundation supporting Inaya Network's long-term growth.

SECTION 12

Conclusion

> *Absolute Protection, Vigilant Care, and Functional Grace — the guiding principles of a network built to guard what matters most.*

The Inaya Protocol demonstrates that data integrity doesn't require a centralized database — it can be achieved through client-side encryption, decentralized storage, and a public, tamper-evident ledger. Everything described in this document runs today on BNB Chain Testnet.