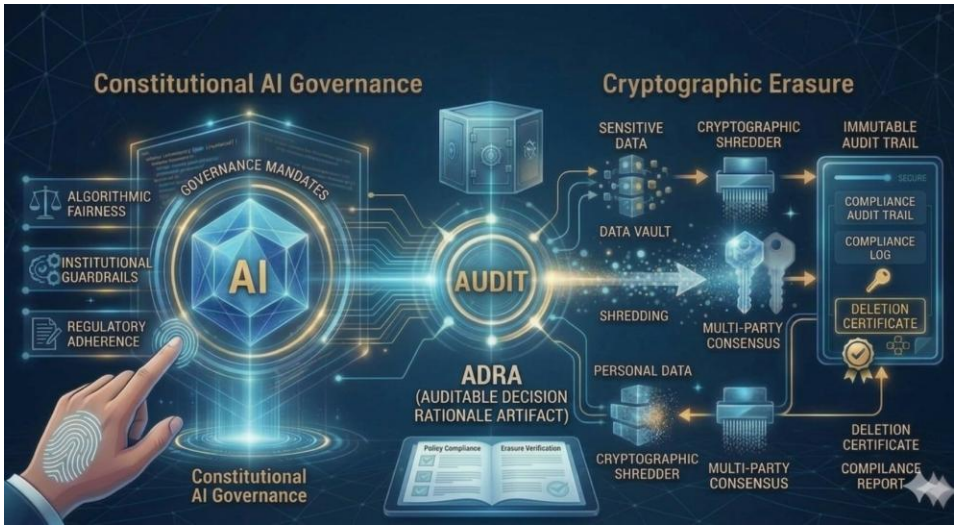
Constitutional AI Governance Meets Cryptographic Erasure



Two Halves of One Audit, Joint Integration Reference

Item	Detail
Version	5
Date	20 August 2026
Authors	Valentine Nsukuzonke (GNCE Labs) · Dylan Wolpe (AT-1 / TinyFiles)
Status	Complete — both halves complete; cross-verified both directions (August 2026); staging validation complete (ledger_record_verified: true) Regression gate: 722/722 (August 2026 estate).
Artifacts (synthetic)	integrations/at1_gnce_bundle/ — pinned CLI @tinyfiles/cli@0.1.81, 4 scenarios, certificates, L8 witnesses, chain export
Artifacts (staging)	https://tinyfiles.io/api/v1 — engine 0.2.34 (Aug 2026 run, issuer_key configured), 0.2.33 Phase 2 baseline, schema 1.2, ledger_record_verified: true, issuer_key: 8cf284cc...
Citation reference	Nsukuzonke, V. & Wolpe, D. (2026). *Two Halves of One Audit: Constitutional AI Governance Meets Cryptographic Erasure.* GNCE Labs × AT-1 integration reference.

Table of Contents

Executive Summary

Problem. Regulated organizations must satisfy two demands at once: maintain tamper-evident audit trails (SOC 2, ISO 27001, sector supervision) and honor data-subject erasure rights (POPIA Section 14, GDPR Article 17). Conventional architectures force a trade-off — keep everything immutable and risk non-compliance with erasure law, or delete data and break the integrity proof.

Approach. Separate decision proof from payload content. GNCE records *what was decided* and *H(payload)* in an immutable sovereign ledger receipt — without storing personal data on the chain. AT-1 holds payload content in a per-subject erasable archive. When erasure is required, AT-1 destroys the subject key (crypto-shredding) and returns an unsigned erasure certificate with canonical wire bytes proving the archive was not tampered with (archive_hash_before == archive_hash_after). GNCE DPO signs those bytes locally (the private key never leaves GNCE custody); AT-1 provides mechanism, not governance authority. GNCE then seals the signed certificate as a supplementary L8 witness on its constitutional hash chain, cross-linked to the original sars_id and payload_hash. Delivery to AT-1 is async — evaluation never blocks on archive I/O.

Demonstrated outcomes (August 2026). Rows 1–4: *pinned synthetic bundle (@tinyfiles/cli@0.1.81, citable fixture hashes). Row 5: hosted staging API (production-shaped async path; run-specific IDs).*

Scenario	Regulatory context	GNCE verdict	Erasure result
UAE healthcare — missing consent	Healthcare consent / warrant	REQUIRE_WARRANT	Certificate VALID; archive unchanged; L8 witness sealed
SA fintech — Google Cloud ZA	Cross-border + POPIA retention	ALLOW	Certificate VALID; archive unchanged; L8 witness sealed
Healthcare PHI — Azure US	HIPAA + CLOUD Act	DENY + veto	Certificate VALID; archive unchanged; L8 witness sealed
ZA banking — memory-only evaluation	In-memory governance check	ALLOW	NO_ERASURE_OBLIGATION; no AT-1 archive, cert, or L8
Healthcare PHI —	HIPAA + CLOUD Act	DENY + veto	Staging ingest 201 → erasure 200 → GNCE DPO sign → L8

Scenario 3 intent (staging re-run, live API)			seal; archive_unchanged: true; signature_valid: true; ledger_record_verified: true; issuer_key present in signed certificate; ERASURE_WITNESSED

Note: Row 5 uses the same governance story as row 3 on Dylan Wolpe's hosted staging (https://tinyfiles.io/api/v1, schema 1.2). It does not replace rows 1–4 for public citation — cite manifest.json for the bundle and run-specific sars_id / archive_ref for staging (Part IV, row 5).

Key finding. In the DENY + veto case, execution was constitutionally blocked, yet erasure still completed cleanly. **Erasure obligation follows persisted content, not whether the action proceeded** — critical for healthcare, banking, and any domain where blocked intents still touch personal data.

Re-run validation (August 2026). A second full bundle execution produced sequential chain positions constitutional_hash with fresh hashes — proving the integration is **stateless and reproducible**, not dependent on cached first-run artefacts. Scenario 4 closes the audit loop: when content is never persisted, GNCE declares `NO_ERASURE_OBLIGATION` with no AT-1 archive, certificate, or L8 witness.

Staging validation (9 August 2026): GNCE connected to AT-1's hosted staging API (tinyfiles.io/api/v1, schema 1.2). A full (non-mocked) evaluation of the healthcare PHI scenario (evaluate_intent → SARS on ledger) was followed asynchronously by worker delivery of POST /archives and POST /archives/{ref}/erasures, local GNCE DPO signing, and L8 supplementary witness with ledger_record_verified: true and erasure_status: ERASURE_WITNESSED. Scenario 3's DENY+veto + erasure story is now demonstrated on live infrastructure, not only the synthetic bundle.

Trust model. GNCE holds the trust root (GNCE DPO signing key). AT-1 provides mechanism only — erasable storage and certificate format — and does not co-sign governance decisions. Both halves bind via `payload_hash`.

Metering is tied to the constitutional evaluation itself, not to the verdict. Every primary SARS receipt — whether ALLOW, DENY, or REQUIRE_WARRANT — represents a governance event. This aligns cost directly with value: you pay for the enforcement, not for the outcome.

Audit artefacts available. ADRA manifests, AT-1 erasure certificates, L8 witness JSON, reproducible CLI pipeline (run_all.ps1), machine-verifiable regression gate (722/722 tests, August 2026 estate). AT-1 CLI pinned to @tinyfiles/cli@0.1.81 in the bundle manifest.

For regulators. This is compliance engineering, not policy documentation. Every claim in this paper maps to reproducible CLI output and JSON artefacts in the repository.

Abstract

Most compliance architectures treat decision-making and data retention as separate problems. Regulators ask for both: prove the action was lawful, and prove personal data was retained correctly and erased on request. Those demands collide when audit systems require immutability and privacy law requires deletion.

GNCE and AT-1 demonstrate a third path: the audit trail continues; the personal data vanishes; and the ledger records both facts cryptographically.

• **GNCE** produces an immutable ADRA receipt — verdict, constitutional articles, H(payload), timestamp — without storing payload content in the ledger.

• **AT-1** holds payload content in a tamper-evident, per-subject erasable archive. When erasure is required, AT-1 destroys the subject key (crypto-shredding) and returns an **unsigned** erasure certificate with canonical wire bytes proving the archive was not tampered with (archive_hash_before == archive_hash_after). **GNCE DPO** signs those bytes locally (the private key never leaves GNCE custody); AT-1 provides mechanism, not governance authority.

• **GNCE L8** seals the **DPO-signed** certificate as a supplementary hash-chain witness, cross-linked to the original sars_id and payload_hash. Delivery to AT-1 is async — evaluation never blocks on archive I/O.

This document is structured as a joint paper: Part II (GNCE) and Part III (AT-1) are both complete; the integration is cross-verified in both directions, including staging validation against Dylan's AT-1 endpoint.

Joint framing — erasure obligation lifecycle

Both authors agree: **persistence is an explicit state, not an inference.** The erasure obligation has a provable lifecycle, fully decoupled from the GNCE verdict.

State	Trigger	Meaning
`NO_ERASURE_OBLIGATION`	Payload never left memory (`payload_content_persisted: false`)	No content store → nothing to erase, nothing to prove — stated explicitly, not silently assumed
`ERASURE_PENDING`	Content persisted to operational store (e.g. AT-1)	Obligation exists; erasure not yet witnessed on the hash chain
`ERASURE_WITNESSED`	AT-1 erasure certificate + GNCE L8 seal	Terminal state — earned, not asserted

PENDING → WITNESSED is not a label change. The AT-1 certificate is the exact trigger: Ed25519 signature over the canonical cert body, `archive_hash_before == archive_hash_after`, and `payload_hash` bound to the original ADRA receipt. GNCE L8 seals the certificate hash and archive integrity into a supplementary constitutional witness.

> **"The obligation follows the data, not the decision."* — joint integration principle (validated by both authors, August 2026)*

Scenario 3 proves this under DENY + veto. Scenario 4 proves the honest negative: memory-only evaluation → `NO_ERASURE_OBLIGATION`, no AT-1 artefacts.

Part I — The Contradiction

1.1 POPIA / GDPR vs immutable audit

POPIA Section 14 and GDPR Article 17 require erasure on request. SOC 2, ISO 27001, and sector regulators simultaneously require tamper-evident audit trails. Most stacks resolve this by:

1. Keeping everything forever and hoping deletion is never demanded, or
2. Deleting on request and breaking the integrity proof.

Both fail under scrutiny. The regulator wants **proof of governance** and **proof of erasure** in the same audit window.

1.2 The separation that makes both possible

What	Treatment
Decision proof	Immutable — verdict, articles, `H(payload)`, witness
Payload content	Erasable — operational store, subject-scoped keys
Erasure proof	Signed certificate — archive untouched, key destroyed

GNCE never stores payload content in the sovereign ledger. AT-1 never attests constitutional verdicts. Each system owns one half; the handoff is `payload_hash`.

Part II — GNCE Half: The Decision Record

Author: Valentine Nsukuzonke / GNCE Labs

2.1 Purpose (regulator & auditor view)

GNCE answers one question for regulators, auditors, and compliance officers: **Was this data processing action lawful under applicable rules at the moment of evaluation?**

It produces a machine-verifiable receipt (ADRA) with verdict, violated articles, and a cryptographic commitment to the payload (payload_hash) — without storing personal data in the audit ledger.

The sovereign ledger records governance over a processing event, not retention of the underlying record. Operational payload content, when persistence is required, is held in a separate erasable store ***(AT-1 — see Part III).*** GNCE binds the two halves through payload_hash, sars_id, and supplementary L8 witnesses.

2.2 Evaluation pipeline (L0–L8)

Layer	Regulatory relevance
L0a	Raw intent capture — stable SHA-256 hash and truncated preview of the inbound request *before* normalization; binds original request for audit and provenance
L0	Trust boundary — sovereignty verification, payload normalization, optional payload-signature policy; includes **L0b caller capability enforcement** when configured
L0b	*(within L0, optional)* Caller capability enforcement — origin_system → allowed_actions
L2a	Intent Consistency Check (ICC) — raw preview vs declared action/risk; CRITICAL veto in CONSTITUTIONAL mode
L2b	Shadow classification — analytics-only inferred intent
L1–L3	Which laws/regimes apply; policy evaluation; decision engine
L4–L5	Per-regime audit artifact (ADRA); evaluation metadata
L6	Behavioural drift monitoring (non-blocking)
L7	Immutable ledger witness (PRECOMMIT → FINALIZE)
L8	Supplementary witnesses (including erasure proof)

Each layer is independently auditable. At integration time the regression gate stood at **722/722** automated tests **GNCE suite only**, including L8 erasure witness sealing, the data-persistence contract, and AT-1 handoff queue/worker behaviour.

Execution order: L0a → L0 → L1–L3 → L4–L5 → L7 → L8 (L6 is observational and non-blocking). Archive ingest, erasure delivery, DPO signing, and L8 erasure witnessing run **asynchronously** when content is persisted to an operational store — constitutional evaluation through L7 never blocks on external archive I/O.

L0 sub-steps (inside the trust boundary). Regulator-facing tables use **L0a** and **L0** as the two trust-boundary labels. Implementers may also refer to **L0b** — an optional sub-step *within* L0, not a separate numbered layer:

Sub-step	Meaning
L0a	Raw intent hash + preview; optional HMAC payload-signature verify and policy
L0 (core)	Sovereign security enforcer; strict JSON normalization; sovereignty verifier — fail-closed in CONSTITUTIONAL mode
L0b	**Caller capability enforcement** — when configured, restricts origin_system / caller to an explicit allowed_actions list; denies undeclared high-risk actions before routing (layer tag: L0_CAPABILITY_ENFORCEMENT)

L0b applies only when caller capability maps are deployed (e.g. enterprise gateway, multi-tenant pilot). Evaluations without that configuration proceed through L0a + L0 core only; absence of L0b is not a gap in the audit trail.

Design principle: evaluate_intent completes synchronously through L7. Archive ingest, erasure delivery, DPO signing, and L8 erasure witnessing run **asynchronously** when content is persisted to an operational store — so constitutional evaluation never blocks on external archive I/O.

2.3 What the ADRA receipt contains

Field	Regulator and Auditor use
Verdict	ALLOW / DENY / REQUIRE_WARRANT
Violated articles	Which rules were breached or satisfied
Primary blocking authority	Which regime/article blocked processing
payload_hash	Proof of which payload was evaluated (content not stored)
SARS ID	Ledger reference for independent verification

Personal data does not enter the sovereign ledger. Regulators and Auditors can verify *what was decided* and *which payload commitment was evaluated* without GNCE retaining subject-level content in the tamper-evident chain.

When processing is blocked (sovereign veto)

If GNCE issues **DENY** with sovereign veto, execution stops. The receipt still documents what was evaluated, which articles fired, and the payload commitment.

Critical for healthcare, banking, and any domain where evaluation touches personal data: if payload content was persisted to an operational store before or during the evaluation path, **erasure law may still apply** — independent of whether the action was allowed to proceed.

Scenario 3 in Part IV demonstrates this explicitly: HIPAA_CE veto blocks execution, yet the persisted operational copy carries a full erasure obligation that completes with a valid certificate and L8 witness. **Erasure obligation follows persisted content, not verdict.**

2.4 Data persistence and erasure status

Every evaluation declares persistence scope explicitly in evaluation metadata. Auditors always see a declared status — no silent inference.

Erasure status	Meaning for Regulators and Auditors
NO_ERASURE_OBLIGATION	Payload evaluated in memory only; no operational content store
ERASURE_PENDING	Content persisted to operational store (e.g. AT-1); erasure not yet witnessed on hash chain
ERASURE_WITNESSED	AT-1 erasure certificate received, GNCE DPO signature verified, L8 supplementary witness sealed

PENDING → WITNESSED is not a label change. It is earned when: (1) AT-1 returns an unsigned erasure certificate with archive_hash_before == archive_hash_after, (2) GNCE DPO signs the exact wire bytes AT-1 supplied, and (3) L8 seals a supplementary witness bound to the original sars_id, adra_id, and payload_hash.

Scenario 4 (Part IV) is the control case: memory-only evaluation
→ NO_ERASURE_OBLIGATION with no AT-1 archive, certificate, or L8 artefact.

L8 erasure witness

When erasure occurs, GNCE appends a **supplementary** hash-chain entry. Primary SARS records finalized at L7 are never modified.

Each L8 erasure witness binds:

- Original sars_id and adra_id (cross-linked to the L7 decision record)
- Same payload_hash as the ADRA receipt
- AT-1 erasure certificate fields, including archive_hash_before == archive_hash_after
- at1_cert_hash — SHA-256 of the canonical signed certificate body
- GNCE DPO signature verification result (signature_valid)
- ledger_record_verified — confirmation that the witness bindings match the live ledger row for that sars_id

Auditor anchor: cite payload_hash equality across ADRA receipt, AT-1 certificate, and L8 witness; cite integrity_hash / constitutional_hash for chain lookup rather than chain position alone (positions increment on each seal).

Example witness fields (staging validation run, August 2026):

Field	Example value
type	AT1_ERASURE_WITNESS
erasure_status	ERASURE_WITNESSED
ledger_record_verified	true
archive_unchanged	true
signature_valid	true

2.5 AT-1 production handoff (async operational store)

As of schema **v1.2** (August 2026), GNCE delivers operational-store ingest and erasure to AT-1 through a **non-blocking file queue and background worker**. This is the production-shaped path validated against Dylan Wolpe's hosted staging API (https://tinyfiles.io/api/v1, engine 0.2.33).

End-to-end flow

```
Intent payload
    → evaluate_intent()                    [synchronous: L0–L7, SARS on ledger]
    → persistence_scope = operational_store?
        → enqueue gnce.at1.archive.ingest   [non-blocking; returns nexus immediately]
    → run_at1_handoff_worker (background)
        → POST /v1/archives                 [ingest; index sars_id → archive_ref]
        → POST /v1/archives/{ref}/erasures [when obligated]
        → GNCE DPO sign unsigned_certificate_bytes locally
        → L8 supplementary witness seal     [same GNCE_LEDGER_DIR as evaluation]
```

Trust boundary (Part 3.4): AT-1 returns **unsigned** canonical certificate bytes. GNCE DPO signs locally with an Ed25519 key that never leaves GNCE custody. AT-1 provides crypto-shredding mechanism; GNCE provides governance authority and ledger sealing.

Handoff envelope (GNCE-side)

Queue messages carry schema_version:
"1.2", event_type, idempotency_key, at1_request (method, path, body),
and adra_manifest (SARS bindings for signing and L8). The worker unwraps GNCE envelope fields; **AT-1 receives ingest/erasure HTTP bodies only.**

Event type	HTTP action	Idempotency key
gnce.at1.archive.ingest	POST /v1/archives	{sars_id}:ingest
gnce.at1.archive.erasure	POST /v1/archives/{ref}/erasures	{sars_id}:erasure
gnce.at1.subject.dsar_erasure	Fan-out erasure across all archives for subject_id	{subject_id}:dsar_erasure (queue); {sars_id}:erasure per HTTP call
gnce.at1.memory_only.ack	Audit log only	{sars_id}:memory_only

HTTP requests include header X-AT1-Schema-Version: 1.2 .

Subject-level DSAR

Erasure obligation follows the **person**, not a single evaluation. GNCE maintains a subject → archive index (archive_index.json). A DSAR enqueue resolves all archive_ref values for a subject_id, erases each through AT-1, signs and L8-seals per archive, and raises an explicit alert (DSAR_ERASURE_PARTIAL) if any archive fails — because partial erasure is a distinct compliance failure from total failure.

Failure handling

Failure class	Location	Alert
Ingest failure	dlq/	Standard dead-letter
Erasure failure	dlq_erasure/	ERASURE_OBLIGATION_UNFULFILLED
DSAR partial	dlq_erasure/alerts/	DSAR_ERASURE_PARTIAL

Erasure failures carry an ageing clock — unfulfilled obligations are visible to operators and auditors, not silently dropped.

2.6 Staging validation (August 2026)

Prior integration proof used the synthetic bundle in integrations/at1_gnce_bundle/ (reproducible CLI, pinned @tinyfiles/cli@0.1.81). In August 2026 GNCE additionally validated the hosted staging deployment:

1. Full, non-mocked evaluate_intent on the healthcare PHI scenario (DENY + sovereign veto, HIPAA_CE).
2. Async enqueue → worker POST /archives → **201**, state ERASURE_PENDING.
3. Worker POST /archives/{ref}/erasures → **200**, unsigned certificate returned.

4. GNCE DPO local sign → L8 seal on the **same ledger directory** used by the evaluation.

Observed success markers: archive_unchanged: true, signature_valid: true, ledger_record_verified: true, erasure_status: ERASURE_WITNESSED.

This closes the gap previously explained by bundle-isolated evaluation (ledger_record_verified: false on synthetic witnesses). Production-shaped runs use one ledger for evaluate and seal; the constitutional hash chain binds decision proof to erasure proof end to end.

Citation notes: Bundle scenarios 1–4 remain the pinned reproducible artefacts for public citation. Staging runs produce fresh sars_id and payload_hash values each execution — cite run-specific identifiers, not bundle fixture hashes.

2.7 Governance guarantees

1. **No payload bypass** — all evaluations route through the jurisdiction router and constitution catalog; regime and article lists are never hardcoded for scenarios.
2. **Ledger boundary** — GNCE submits; the ledger seals. GNCE does not self-attest finalization.
3. **Constitutional veto** — overrides the automated decision engine when rules require; veto is recorded on the receipt.
4. **Strict canonical JSON** — cryptographic proofs use deterministic serialization (stable_json_strict); verifiers reproduce signed bytes without re-inference.
5. **Explicit persistence contract** — every evaluation declares persistence_scope and erasure status; memory-only paths emit NO_ERASURE_OBLIGATION explicitly.
6. **Separation of decision proof and content** — payload content in operational store; governance proof on ledger; binding via payload_hash only.

2.8 Technical reference (implementers)

Layer map

Layer / step	Role
L0a	**Raw intent capture; optional payload-signature verify and policy**
L0	**Sovereign security enforcer, strict normalization, trust boundary (fail-closed in CONSTITUTIONAL mode)**
L0b	***(within L0, optional)* Caller capability enforcement — origin_system → allowed_actions**

L2a	**Intent Consistency Check (ICC) — raw preview vs declared action/risk; CRITICAL veto in CONSTITUTIONAL mode**
L2b	**Shadow classification — analytics-only inferred intent**
L1–L3	**Jurisdiction router → constitution catalog → policy engines → DRE**
L4–L5	**Per-regime ADRA forging; system metadata**
L6	**Behavioural drift (observational, non-blocking)**
L7	**PRECOMMIT → MINT → SEAL → FINALIZE**
L8	**Supplementary witnesses (including AT-1 erasure proof)**

Implementer module map and staging scripts: *see repository Appendix A (docs/AT1_HANDOFF_SCHEMA.md, ICC_CUSTOMER_EXPLAINER.md).*

2.9 Core modules

Module	Purpose
gnce/gn_kernel/phases/raw_intent_capture.py	L0a — raw intent hash/preview, payload signature verify + policy
gnce/gn_kernel/phases/l0_security_phase.py	L0 — sovereign enforcer, L0b caller capabilities, trust boundary
gnce/gn_kernel/phases/l8_erasure_witness.py	L8 supplementary witness build, verify, seal
gnce/gn_kernel/phases/data_persistence_contract.py	Persistence scope and erasure status contract
gnce/integrations/at1_handoff.py	Post-eval envelope builders, persistence routing, evaluate_and_enqueue_at1_handoff
gnce/integrations/at1_handoff_queue.py	File queue (pending / completed / DLQ), DSAR partial alerts, archive index
gnce/integrations/at1_handoff_worker.py	AT-1 HTTP client, ingest/erasure delivery, DPO sign, L8 seal

gnce/integrations/at1_cert_signing.py	Ed25519 signing over AT-1 unsigned_certificate_bytes (wire-exact)
gnce/integrations/at1_archive_index.py	sars_id / subject_id → archive_ref index for DSAR fan-out

Primary GNCE modules for AT-1 integration (schema v1.2): gnce/gn_kernel/phases/raw_intent_capture.py (L0a), l0_security_phase.py (L0/L0b), data_persistence_contract.py, l8_erasure_witness.py; gnce/integrations/at1_handoff.py, at1_handoff_queue.py, at1_handoff_worker.py, at1_cert_signing.py, at1_archive_index.py. Full module map: Appendix A (Table A.2).

2.10 CLI and validation scripts

Script	Purpose
scripts/seal_at1_erasure_witness.py	Seal bundle certificates as L8 witnesses
scripts/smoke_at1_staging.ps1	Staging smoke: health → ingest → erasure → DPO sign
scripts/run_at1_staging_real_l8.ps1	Real evaluation + staging + full L8 (ledger_record_verified: true)
scripts/run_at1_handoff_worker.py	Worker entrypoint (queue drain)
scripts/verify_l8_witness_bundle.py	Independent cert + witness verification
scripts/verify_l8_chain_export.py	Constitutional hash chain verification export

Key scripts: scripts/seal_at1_erasure_witness.py, smoke_at1_staging.ps1, run_at1_staging_real_l8.ps1, run_at1_handoff_worker.py, verify_l8_witness_bundle.py, verify_l8_chain_export.py, at1_erasure_cert_verifier.py. Reproducible bundle pipeline: integrations/at1_gnce_bundle/run_all.ps1. Full index: Appendix A (Tables A.5–A.6).

2.11 Environment (staging / pilot)

Variable	Purpose
GNCE_AT1_BASE_URL	AT-1 API base (e.g. https://tinyfiles.io/api/v1)
GNCE_AT1_BEARER_TOKEN	Staging authentication
GNCE_AT1_HANDOFF_ENABLED	Enable post-eval enqueue

GNCE_AT1_SIGNING_KEY / GNCE_AT1_PUBLIC_KEY	GNCE DPO Ed25519 key pair
GNCE_AT1_HANDOFF_QUEUE_DIR	File queue root
GNCE_LEDGER_DIR	Sovereign ledger (must be shared for evaluate + L8 seal)
X-AT1-Schema-Version: 1.2	Set by worker on all AT-1 HTTP calls

Schema contract: docs/AT1_HANDOFF_SCHEMA.md (v1.2).

Caller capabilities (L0b) are configured on the kernel instance (_caller_capabilities), not via these AT-1 env vars. Schema contract: docs/AT1_HANDOFF_SCHEMA.md (v1.2). Full L0 sub-step specification: docs/GNCE_L0-L8_Stack_Sanity_Check_PDF.md 10.

End of Part II — GNCE Half.

Part III — AT-1 Half: Cryptographic Erasure

Author: Dylan Wolpe (AT-1 / TinyFiles)

> **Design validation (August 2026).** Dylan confirmed the persistence contract framing: explicit lifecycle states beat inferred obligations; the AT-1 certificate body is the earned trigger for ERASURE_WITNESSED once GNCE DPO has signed it; NO_ERASURE_OBLIGATION makes the memory-only boundary honest rather than silent. Independent verification is in 3.7.

3.1 Design goal

AT-1 is a data shop, not a governance engine. It holds the records a regulated business is obliged to keep, and it has to satisfy two distinct sets of demands. One requires a tamper-evident archive whose integrity can be demonstrated years later to regulators. The other requires that a named person be removed from it on request, within a statutory window, as auditors and compliance officers expect to verify.

Deleting bytes from a sealed archive breaks the integrity proof the first demand relies on. Keeping everything and reporting the deletion as done breaks the second. Most stacks pick one and hope the question never arrives in the same audit.

The way out is to stop treating the record and the person as the same object. The archive is a fixed artefact; a data subject is a key. Erasure removes the key and leaves the artefact alone, so the archive that was required to stay intact demonstrably did, and the subject who was required to be forgotten demonstrably is. Both facts come out of the same file.

The lifecycle in the joint framing above is the contract AT-1 implements against. Persistence is recorded explicitly at ingest, so NO_ERASURE_OBLIGATION is a state AT-1 asserts rather than a gap the auditor has to interpret.

3.2 Crypto-shredding model

Every record group belonging to one data subject, across all tables in the archive, is encrypted under a key unique to that subject: AES-256-GCM, a fresh 12-byte nonce per group, and the subject_id bound in as additional authenticated data. Binding the identity into the AAD means a ciphertext block cannot be silently re-attributed to a different subject; the tag fails first. Payloads are deflated before encryption, which keeps the per-subject layout from inflating the archive.

Keys live in a vault, separate from the archive. Erasure destroys one key and writes a tombstone recording that it was destroyed and when. The ciphertext is never touched. Nothing is rewritten, re-sealed or re-signed, which is what allows archive_hash_before and archive_hash_after to be equal and still honest.

The consequence for a regulator is narrow and checkable. After erasure the subject's ciphertext remains present but is unrecoverable, because the only key that could open it no longer exists. Every other subject in the archive is unaffected and still queryable. Some EU regulator guidance discusses cryptographic key destruction as putting data "beyond use" in certain contexts; treatment varies by jurisdiction, remaining copies, and backup policy. This section describes a technical mechanism only — not a legal conclusion that erasure obligations are met in every deployment.

Choosing this over physical excision is deliberate. Cutting bytes out of a sealed archive invalidates the integrity proof, and an auditor or regulator asking for erasure evidence usually wants the retention evidence in the same breath.

3.3 Erasure certificate

Each erasure emits a certificate body. It names the subject, the time, how many record groups went, a proof that the specific key was destroyed, and the archive hash before and after.

Trust boundary. On the HTTP handoff path (schema v1.2), AT-1 returns an **unsigned** canonical body plus unsigned_certificate_bytes. **GNCE DPO** signs those bytes locally with Ed25519; the private key never leaves GNCE custody. AT-1 provides mechanism, not governance authority. On the local CLI path, the same body is signed with --signing-key gnce-dpo.key.

The signed body uses compact, key-sorted JSON with (",", ":") separators (Part VI). **cert_version 1.1** (CLI 0.1.81+, engine 0.2.33+): the Ed25519 signature covers **every field except signature** — vault state (vault_state_after, vault_version_after), archive anchoring (prev_archive_hash, archive_instance), issuer binding (issuer_key), and payload_hash when present. **Legacy 1.0** certificates (pinned bundle, August 2026) used a fixed seven-field allowlist plus optional payload_hash; the standalone verifier accepts both.

payload_hash is what closes the loop with GNCE. A certificate on its own proves a subject was erased. With the ADRA receipt's H(payload) bound into the signature, it proves the specific

payload the decision was made about was erased. Altering that hash invalidates the signature, so the decision record and the deletion record cannot drift apart.

Example certificate (Scenario 3 — healthcare PHI, DENY + veto; cert_version **1.1**, CLI **0.1.81** local rebuild):

Legacy /1.0 certificates in the pinned bundle (August 2026) still verify with the standalone verifier; they omit archive_instance, vault state, and issuer_key binding.


```
{
  "subject_id": "gnce-phi-subject-us-003",
  "erased_at": "2026-08-12T16:31:09Z",
  "record_groups_erased": 1,
  "archive_hash_before":
"26c99e686987f0128f93bb3b68e44006ff94d05dc2f4ddd75fcbcefa0f6af048",
  "archive_hash_after":
"26c99e686987f0128f93bb3b68e44006ff94d05dc2f4ddd75fcbcefa0f6af048",
  "key_destruction_proof":
"eae567ff93552b63d6ab70537758c54080b17745343c15c47749729fb4657ba9",
  "issuer": "GNCE DPO",
  "payload_hash": "b39b4b4fa39ccf63c4b854066e21d735007e232d1afcffd0de6c54e4936022ae",
  "cert_version": "1.1",
  "issuer_key": "8cf284ccf0e52260d8887a19d95e7701db59ff60f80cb048b581bb7306f11b8a",
  "archive_instance": "arc-a579383f2bed8f0f",
  "prev_archive_hash":
"26c99e686987f0128f93bb3b68e44006ff94d05dc2f4ddd75fcbcefa0f6af048",
  "signature":
"596c28c7471a6607c277debaa7912bd9a2e931bbc5b41d86aebd5134923b3cdc990a223fcc407af59a8482
079288052fa34c2c4b1e1722f705ab1c981bf2eb09"
}
```


Source: *staging run, 12 August 2026. SARS: sars-499156122bb046a7; Archive: arc-a579383f2bed8f0f; issuer_key present and bound.*

The two archive hashes being identical is the load-bearing claim. It is what distinguishes a certificate from an assertion: the archive under audit is byte-for-byte the archive that existed before the erasure, and the subject is still gone.

Under cert_version 1.1, the signed body covers all fields present. This includes vault_state_after and vault_version_after when emitted. Hosted staging deliberately omits these fields by design — the API holds one key per archive and has no vault in the engine's sense. Absence is honest, not a defect.

Scope note: *This applies to default crypto-shred mode. Under* --mode redact*, archive hashes change; the guarantee shifts to survivor roots instead. GNCE will add a dedicated* others_unchanged *field before sealing redact-mode erasure witnesses into L8, ensuring the witness records the correct invariant (survivor-root equality) rather than the archive hash change. (AT-1 uses* key-only *internally; GNCE witnesses record this as* crypto_shred*.)*

3.4 Trust boundary: mechanism, not authority

AT-1 holds no signing authority in this integration. The certificate is signed with the GNCE DPO key, generated and held by GNCE. AT-1 supplies the erasable store, the unsigned certificate body and wire bytes, the certificate format, and the verification tool — and signs nothing.

This is not modesty about the product. If the vendor whose software performed the erasure also attested that it happened, the attestation would be worth very little to an auditor or regulator. Separating them means the evidence survives someone doubting AT-1, which is the condition it has to meet.

Verification follows from the same principle. at1_erasure_cert_verifier.py ships as a standalone file that contains no AT-1 engine internals and depends only on a public Ed25519 and SHA-256 implementation. An auditor, a regulator, or GNCE's own L8 phase can run it without installing AT-1 at all, and a competent engineer can reimplement it from Part VI in an afternoon.

For century-tier retention the certificate supports an optional dual signature, Ed25519 alongside SLH-DSA (FIPS-205), shipped in AT-1 **engine 0.2.33**, so certificates written today remain verifiable if Ed25519 is later broken. The Ed25519-only path is unchanged by this, and existing certificates keep verifying.

3.5 CLI reference

Versioning. Install the npm CLI with @tinyfiles/cli@0.1.81 (or current 0.1.x on npm). Do not pin npm to engine semver — hosted staging reports engine 0.2.33 and schema 1.2 via GET /version. HTTP requests from the GNCE worker send header X-AT1-Schema-Version: 1.2 (read by the API as x-at1-schema-version). Engine versioning note: engine 0.2.33 is the Phase 2 baseline (pinned bundle and worker contract); engine 0.2.34 is the August 2026 hosted staging run cited in Part IV row 5 (cert_version 1.1 with issuer_key). Both use schema 1.2.

Four commands cover the local bundle integration:

at1 erase keygen --out gnce-dpo.key
Generates the DPO keypair and writes gnce-dpo.key.pub. The private key never leaves GNCE.

at1 erase build <scenario.at1.json> --subject-field subject_id --hash-field payload_hash --out <archive_dir>/
Builds the erasable archive and binds the caller's payload_hash into the archive metadata.

at1 erase erase <archive_dir>/ <subject_id> --signing-key gnce-dpo.key --issuer "GNCE DPO" --out-cert <cert.json>
Crypto-shreds the subject key. On the CLI path, GNCE DPO signs locally and writes the certificate (payload_hash in the signed body when present). On the HTTP path, AT-1 returns unsigned bytes; GNCE signs in the worker.

at1 erase verify <cert.json> <gnce-dpo.key.pub>
Checks the Ed25519 signature and the stated guarantees, printing VERIFIED or REJECTED with matching exit codes (0 / 1). A distinct exit code 3 signals an unsupported format rather than a

verdict. Verification establishes that the certificate is authentic and internally consistent; whether it satisfies a particular legal obligation is a determination for the relevant authority.

Note (August 2026): The CLI at1 erase verify command on engine 0.2.33+ aligns with scripts/at1_erasure_cert_verifier.py (combined signature-and-binding checks for /1.0 and /1.1). For byte-for-byte proof, run: py scripts/at1_erasure_cert_verifier.py <cert.json> gnce-dpo.key.pub (same script ships in integrations/at1_gnce_bundle/at1_erasure_cert_verifier.py).

The same script ships in integrations*/at1_gnce_bundle/at1_erasure_cert_verifier.py.*

3.6 Threat model

Stated as what each party can do, and what stops them.

Adversary / failure	Attempt	What prevents it
Archive is exfiltrated	Read an erased subject's records	Ciphertext only. The per-subject key is destroyed, so the block cannot be opened.
Operator fakes an erasure	Report deletion without performing it	The certificate names a destroyed key and a tombstone; the vault has no key to produce.
Operator quietly edits the archive	Remove or alter other records while erasing one	`archive_hash_after` no longer equals `archive_hash_before`; the certificate fails verification.
Certificate substitution	Swap in a certificate from another erasure	The L8 witness commits to the SHA-256 of the canonical signed body; re-serialising does not change it.
AT-1 (the vendor)	Attest an erasure that did not happen	AT-1 holds no signing key. Only the GNCE DPO key produces a valid certificate.
Future cryptanalysis	Forge a signature years later	Optional SLH-DSA dual-signature path for century-tier retention.

Two limits belong in the record rather than in a footnote. First, `key_destruction_proof` attests that the vault destroyed the key; the strength of that attestation is the strength of the vault, so an HSM or KMS-backed deployment is doing real work that a software vault is not. Second, erasure reaches only the archives and vault replicas within its scope. A key surviving in an out-of-scope backup is a live subject, and any deployment has to bring backup and replica policy into the same boundary. Neither is a cryptographic gap; both are deployment obligations, and a regulator or auditor will ask about them.

3.7 Independent verification (August 2026)

The August 2026 handoff was verified in both directions rather than asserted in one — first on the pinned synthetic bundle, then on hosted staging using the production-**shaped** async path (Part II; Part IV, row 5).

Synthetic bundle (reproducible proof)

GNCE's synthetic bundle was checked against AT-1's reference verifier (at1 erase verify / scripts/at1_erasure_cert_verifier.py): all three persisted scenarios returned a valid GNCE DPO signature over the canonical certificate body, with archive_hash_before equal to archive_hash_after. Each L8 witness's payload_hash matched its certificate, and each witness's at1_cert_hash matched the SHA-256 of that certificate's canonical signed body. Scenario 4 produced no certificate, as the contract requires.

Bundle certificates are cert_version 1.0 (August 2026 pinned fixtures); see 3.3 for the current 1.1 format.

The constitutional hash-chain export was verified independently by reimplementing GNCE's sealing rule rather than trusting the supplied script alone. For each witness, the entry's integrity_hash recomputed from its documented metadata; previous_hash linked to the prior entry; and that prior entry's hash recomputed as well. Lookup is anchored by hash rather than chain position, which matters because legacy chains may contain duplicate positions — auditors should cite integrity_hash / constitutional_hash, not position alone.

The published integrity_hash_algorithm block in integrations/at1_gnce_bundle/l8_chain_verification_export.json now includes a defaults_when_absent map (regime → "UNKNOWN", decision → "◯ UNKNOWN", adra_id falling back to sars_row_id when present), so an independent verifier can recompute seals from the export alone without reference to GNCE source. Regenerate with:

py scripts/verify_l8_chain_export.py --export integrations/at1_gnce_bundle/l8_chain_verification_export.json

On ledger_record_verified: false in bundle witnesses: this flag on scenarios 1–3 reflects **bundle-isolated evaluation** — ADRA manifests and L8 seals were produced without a shared live ledger row from a preceding evaluate_intent in the same process. It is an artefact of the reproducible fixture path, not a cryptographic defect. Certificate signature, payload_hash binding, at1_cert_hash, and hash-chain linkage still verify independently.

Hosted staging (wire-contract validation)

In August 2026, GNCE validated Scenario 3's intent against Dylan Wolpe's hosted staging API (https://tinyfiles.io/api/v1, **engine 0.2.34**, **schema 1.2**). **Synthetic/anonymised payloads only.** This environment is explicitly **non-production**: per-instance storage, no durable shared database, no HSM, no SLA — as documented by AT-1 for Phase 1.

A real evaluate_intent wrote a SARS row to the ledger; a background worker delivered POST /v1/archives and POST /v1/archives/{ref}/erasures with header **X-AT1-Schema-Version: 1.2**; GNCE DPO signed AT-1's unsigned_certificate_bytes locally; and L8 sealed a supplementary witness using the same GNCE_LEDGER_DIR.

Observed success markers: archive_unchanged: true, signature_valid: true, ledger_record_verified: true, erasure_status: ERASURE_WITNESSED.

This closes the **wire-contract / shared-ledger witness gap** on staging: when evaluation and L8 sealing share one ledger, the witness bindings match the live SARS record for that sars_id and adra_id. It does **not** certify production infrastructure, sector pilots, or regulatory approval.

Early August 2026 staging runs (11 Aug) returned cert_version 1.0 and proved the async handoff path. From 12 August 2026, live staging on engine 0.2.34 issues cert_version 1.1 with archive_instance, prev_archive_hash, and (after DPO key configuration) issuer_key — all bound in the signed body. Legacy /1.0 certificates still verify with the standalone verifier and remain valid for the pinned bundle. Phase 2 baseline remains CLI 0.1.81 / engine 0.2.33; the 12 Aug run is the separate live proof = *August staging run cited in row 5* engine 0.2.34 (queue 183052).

Staging runs produce **fresh identifiers** each execution (sars_id, archive_ref, payload_hash) — cite run-specific JSON, not bundle fixture hashes in manifest.json. Published exemplar: integrations/at1_gnce_bundle/certificates/03_healthcare_phi_deny_erasure_cert_staging_20260811.json (full queue artefact: .gnce/at1_real_l8/queue_20260811_113339/).

Example certificate (staging run, 11 August 2026 — synthetic; run-specific IDs):
sars_id: sars-64ad905e99274f14 · archive_ref: arc-6eaaab1728e29b81 · adra_id: 027937608593463884 · verdict: **DENY + sovereign veto (Scenario 3 — healthcare PHI; bundle ADRA cites HIPAA §164.308)**


```
{
  "subject_id": "gnce-phi-subject-us-003",
  "erased_at": "2026-08-11T09:33:42Z",
  "record_groups_erased": 1,
  "archive_hash_before":
"a190c84726adf5afa4e323ed1a696b14e9191f044f0fb7deb5259b5503b4cc47",
  "archive_hash_after":
"a190c84726adf5afa4e323ed1a696b14e9191f044f0fb7deb5259b5503b4cc47",
  "key_destruction_proof":
"479e398d6e74512414a04935b8314d3c038eb6a6e29c53b3d4222626ae384545",
  "issuer": "GNCE DPO",
  "payload_hash": "42cb15d9552f10d71d6861fd210f8c96ba00abc70dba04d28e5f9e981411c88c",
  "signature":
"44e1cacf2aa41949d15db256a2c71d97dcf2e069c6bd3bb7cc41a6d582c87cdd06596c0c0e883ef921a34e
d548157b7ae93224007954456b3201816644093c02"
}
```


Source: integrations/at1_gnce_bundle/certificates/03_healthcare_phi_deny_erasure_cert_staging_20260811.json (exemplar from run queue_20260811_113339; full queue JSON under .gnce/at1_real_l8/).

Staging validation with issuer_key **(12 August 2026).** Following configuration of the GNCE DPO public key, a further staging run was performed against https://tinyfiles.io/api/v1 (engine 0.2.34). The resulting certificate now carries issuer_key: 8cf284cc..., binding the issuer field to the signing key. The standalone verifier returns VERIFIED, exit 0 on the reassembled certificate. This closes the final integration item.

Conclusion

The path from AT-1 erasure response → GNCE DPO signature → L8 supplementary witness → constitutional hash chain verifies end to end, with neither side's tooling required to trust the other's. The **bundle** proves reproducible cryptography and binding invariants; **staging** proves the async handoff API and ledger-verified L8 seal on hosted infrastructure — **synthetic data only**, not live End Client production.

Disclaimer. *This paper documents a jointly verified technical integration and reproducible artefacts. Each deployment organization remains responsible for its own legal and sector compliance.*

End of Part III— AT-1 Half.

Part IV — Joint Integration: *Four Scenarios + Staging Validation*

Both authors ran the same synthetic bundle end-to-end on @tinyfiles/cli@0.1.81 (pinned in manifest.json) with GNCE DPO as signing key. Scenarios 1–4 are the reproducible, citable proof — fixed fixture hashes, safe for public citation and CI.

4.1 Staging Validation: Hosted AT-1 Proof (Row 5)

In August 2026, GNCE additionally validated Scenario 3's intent against Dylan Wolpe's hosted staging API (https://tinyfiles.io/api/v1, schema 1.2). Row 5 records the live run: same governance story (healthcare PHI, DENY + sovereign veto), production-shaped async path, and fresh sars_id / archive_ref / payload_hash each execution — do not merge into bundle manifest hashes. Early staging (11 Aug) proved the handoff on cert_version 1.0; from 12 August 2026, live staging on engine 0.2.34 issues cert_version 1.1 with archive_instance, prev_archive_hash, and issuer_key bound in the signed body (configured run: sars-499156122bb046a7, arc-a579383f2bed8f0f, queue 20260812_183052). Phase 2 baseline remains CLI 0.1.81 / engine 0.2.33; engine *0.2.34 = August staging run cited in* Row 5 is the separate live proof.

Scenario table

#	Scenario	GNCE verdict	Subject ID	Persistence	Result
1	UAE healthcare — missing consent	REQUIRE_WARRANT	patient_record_12345	operational store	cert VALID; archive unchanged; L8 witness sealed
2	SA fintech — Google Cloud ZA mitigated	ALLOW	gnce-fintech-subject-za-001	operational store	cert VALID; archive unchanged; L8 witness sealed
3	Healthcare PHI — Azure US, provider keys	**DENY + veto**	gnce-phi-subject-us-003	operational store	cert VALID; archive unchanged; L8 witness sealed
4	ZA banking — low-value transfer	ALLOW	gnce-banking-memory-eval-004	**memory only**	NO_ERASURE_OBLIGATION; no AT-1 archive, cert, or L8
5	Healthcare PHI — Scenario 3 intent (**staging re-run**, live API)	**DENY + veto**	gnce-phi-subject-us-003	operational store (hosted AT-1)	Staging ingest **201** → erasure **200** → DPO sign → L8 seal; archive_unchanged: true; signature_valid: true; ledger_record_verified: true; issuer_key present; erasure_status: ERASURE_WITNESSED

Anchors (12 August 2026 configured run): sars_id: sars-499156122bb046a7; archive_ref/instance: arc-a579383f2bed8f0f; issuer_key: 8cf284cc...

How to read row 5 vs rows 1–3

	Rows 1–3 (bundle)	**Row 5 (staging)**
Proof type	Reproducible synthetic fixtures	Production-shaped live API
AT-1 surface	Local CLI archive (at1 erase build/erase)	POST /v1/archives + POST /v1/archives/{ref}/erasures
Evaluation path	Pre-generated ADRA manifests	Real evaluate_intent → SARS on ledger → async worker
What to cite	Bundle payload_hash in manifest.json	Run-specific sars_id, archive_ref, L8 witness JSON
CLI / engine	@tinyfiles/cli@0.1.81 (pinned)	Staging engine **0.2.33**, schema **1.2**

- Example staging anchors (August 2026 run): sars_id: sars-499156122bb046a7 archive_ref: arc-a579383f2bed8f0f; payload_hash: b39b4b4fa39ccf63c4b854066e21d735007e232d1afcffd0de6c54e4936022ae.

These values change each run — cite the run you reproduce, not these literals alone.

4.2 Re-run observations (reproducible infrastructure)

Three properties distinguish a **one-off demo** from **reproducible infrastructure**:

1. Fresh seals prove ledger health. Re-running scenarios 1–3 produced new L8 witnesses with recomputable integrity_hash values and intact previous_hash links. Each evaluation advances the chain; auditors should anchor on constitutional_hash / integrity_hash, not chain position alone.
2. archive_unchanged=true + signature_valid=true is the invariant. Every persisted scenario — DENY, REQUIRE_WARRANT, ALLOW — produces the same cryptographic guarantee: archive bytes never mutated, GNCE DPO signature verifies.
3. CLI version pinning avoids drift. Bundle pins @tinyfiles/cli@0.1.81 for CI (engine 0.2.33 baseline). Row 5 staging validated on engine 0.2.34. Periodic testing against latest catches breaking changes early.

4.3 Handoff shape

```
Intent payload
    → GNCE evaluate_intent()
        → ADRA receipt (verdict, articles, payload_hash, sars_id)
        → [async] payload content → AT-1 archive (operational store)
    → AT-1 erasure
        → Unsigned certificate + wire bytes (GNCE signs locally)
    → GNCE L8 seal
        → Supplementary witness on constitutional hash chain
        → ledger_record_verified: true (row 5; same ledger as evaluation)
```

4.4 Scenario 3 — why it is the keystone

- **Verdict: DENY** — HIPAA_CE veto; sovereign veto applied.
- **Execution: Blocked** — action did not proceed.
- **Payload:** Persisted to operational store for evaluation/audit path.
- **Erasure: Clean** — archive unchanged, certificate VALID.
- **L8: Witness sealed with signature_valid:** true; on staging, ledger_record_verified: true.

Example L8 witness (Scenario 3)


```
{
  "type": "AT1_ERASURE_WITNESS",
  "sars_id": "sars-10e4264d057a4442",
  "adra_id": "009842692955875265",
  "subject_id": "gnce-phi-subject-us-003",
  "payload_hash": "1a4cd467572bf070a18011c80edbd92446b843471cd4214de3a003d230ccb555",
  "at1_cert_hash": "1133f1f49bc62e42ddaf3af08ab50f50baf603ef4ccbbd29c9e2a80042dd134e",
  "archive_unchanged": true,
  "signature_valid": true,
  "constitutional_hash":
"4ac09ff4c85221a934435f49a2c31006a99319c2a42cdca28bcf072c4636ae02",
  "chain_position": 25580,
  "constitutional_sealed_at": "2026-08-07T13:40:25.121Z",
  "seal_payload_hash":
"cefe530e10bad1941e18ecad4ccef513be231fa935dc0ac362ccdf53d4593497",
  "ledger_record_verified": false,
  "issuer": "GNCE DPO",
  "erased_at": "2026-08-07T13:37:36Z",
  "erasure_status": "ERASURE_WITNESSED"
}
```

*Note: `sars_id`, `payload_hash`, and chain positions are run-specific; re-running the bundle generates new values. Hash equality properties (`archive_unchanged`, binding via `payload_hash`) are stable. *

Conclusion: Governance obligation follows the **persisted copy,** not the verdict. Row 5 proves the same story on live hosted infrastructure, not only the synthetic bundle.

4.5 Scenario 4 — the control case (closes the audit loop)

When payload content is **not** persisted to an operational store:

- `payload_content_persisted: false`
- `persistence_scope: memory_only`
- `erasure_obligation: NONE`
- `Erasure status: NO_ERASURE_OBLIGATION`
- No AT-1 archive built, no erasure certificate, no L8 witness

When persistence_scope: memory_only, GNCE declares NO_ERASURE_OBLIGATION explicitly. No AT-1 archive, no erasure certificate, no L8 witness. Auditors see an honest negative — not silent absence of artefacts.

Part V — Trust Model

5.1. Who Attests What?

Question	Answer
Who attests the decision?	GNCE — ADRA receipt + L7 SARS finalize
Who attests erasure?	GNCE DPO key on AT-1 certificate; verified by scripts/at1_erasure_cert_verifier.py
Who holds the trust root?	GNCE — issuer keygen on integrator side
What does AT-1 provide?	Mechanism — erasable archive, crypto-shred, certificate format
What binds the halves?	`payload_hash` — same field in ADRA receipt, AT-1 cert, and L8 witness

Two hash domains, one signed object:

- **`payload_hash`** — content identity the ADRA receipt commits to
- **`archive_hash_*`** — container integrity (unchanged on clean erase)

Both appear in the erasure certificate; both are cross-linked in the L8 witness.

Part VI — Canonical JSON Contract

Cross-system verification requires identical serialization.

6.1 AT-1 certificate signature body

```
json.dumps(body, sort_keys=True, separators=(",", ":"))
```

Body fields: `subject_id`, `erased_at`, `record_groups_erased`, `archive_hash_before`, `archive_hash_after`, `key_destruction_proof`, `issuer`, `payload_hash` (when present).

6.2 GNCE strict JSON (proofs)

All GNCE cryptographic commitments use `stable_json_strict` — no coercion, canonical UTC timestamps, sorted keys.

6.3 L8 witness verification

GNCE ships a scripts/at1_erasure_cert_verifier.py and seals witnesses using the same canonical rules. Byte-for-byte agreement is required between AT-1 verify and GNCE L8 seal.

6.4 L8 witness independent verification (for AT-1 authors)

Dylan (or any auditor) can verify witnesses without re-running the full GNCE pipeline:

```
1. AT-1 standalone certificate verify (Ed25519 over canonical cert body)
at1 erase verify
integrations/at1_gnce_bundle/certificates/03_healthcare_phi_deny_erasure_cert.json
gnce-dpo.key.pub

2. GNCE cross-check: cert signature + witness field binding + seal_payload_hash
py scripts/verify_l8_witness_bundle.py

3. Single witness
py scripts/verify_l8_witness_bundle.py --witness
integrations/at1_gnce_bundle/witnesses/03_healthcare_phi_deny_l8_witness.json
```

Witness JSONs live in `integrations/at1_gnce_bundle/witnesses/`. Each file cross-links `sars_id`, `adra_id`, `payload_hash`, `at1_cert_hash`, and `seal_payload_hash` (deterministic hash of the L8 seal body).

6.5 Dylan verification handoff package

```
py scripts/generate_at1_dylan_handoff.py
```

Produces:

- `integrations/at1_gnce_bundle/dylan_handoff_manifest.json` — SHA-256 index of every auditable artefact
- `integrations/at1_gnce_bundle/dylan_handoff_bundle.zip` — same files packaged for share-out (no private key, no runtime archives)

Part VII — Reproducing the Integration

7.1 Prerequisites

```
npm i -g @tinyfiles/cli@0.1.81
at1 login
at1 erase keygen --out gnce-dpo.key
```

7.2 Generate GNCE bundle (synthetic)

```
py scripts/generate_at1_gnce_bundle.py
```

7.3 Full pipeline

```
.\integrations\at1_gnce_bundle\run_all.ps1
```

This runs: AT-1 build → read → erase → verify → GNCE L8 seal for scenarios 1–3; scenario 4 verifies memory-only ADRA only.

7.4 Staging validation (live API)

The pinned bundle (above) proves reproducible cryptography on local CLI archives. **Hosted staging** validates the production-shaped path: HTTP API, async worker, GNCE DPO signing, and (for the full script) L8 seal against a real SARS ledger row. Staging runs produce **fresh** sars_id, archive_ref, and payload_hash values each execution — cite run-specific output, not bundle fixture hashes in manifest.json.

7.4.1 Prerequisites (staging only)

From repository root (gnce-v2.4.0):

1. **Credentials** — place Dylan's staging file at staging/GNCE_staging_credentials.txt (base URL + bearer token). Do not commit this file.
2. **DPO key pair** — if missing: at1 erase keygen --out gnce-dpo.key (creates gnce-dpo.key.pub in repo root).
3. **Python environment** — same venv used for pytest / bundle generation.
4. **TLS (Windows). If certificate verification fails locally, it is the Windows trust store rather than the endpoint — tinyfiles.io presents a valid publicly-trusted certificate. Update the local root store, or run the worker from WSL or the Python environment used for the test suite.**

Staging endpoint (August 2026): https://tinyfiles.io/api/v1 — engine 0.2.33, schema 1.2 (X-AT1-Schema-Version: 1.2 on worker requests).

7.4.2 Option A — Smoke test (fast connectivity check)

Confirms staging is up and the HTTP surface accepts ingest, erasure, and local DPO signing. **Does not** run evaluate_intent or L8 seal.

cd E:\Documents\gnce\gnce-v2.4.0

.\scripts\smoke_at1_staging.ps1

What it runs: GET /health → GET /version → POST /archives (synthetic ingest, 201) → POST /archives/{ref}/erasures (200) → GNCE DPO sign → archive_unchanged check.

Optional flags:

Flag	Effect
-SkipIngest	Health + version only
-SkipErasure	Ingest only (no erasure/sign)
-CredentialsFile "path\to\credentials.txt"	Override default credentials path

Pass markers: all steps report PASS; final line: *Staging smoke complete (ingest → erasure → DPO sign).*

7.4.3 Option B — Full staging loop (production-shaped proof)

Confirms the complete GNCE half on hosted staging: full, non-mocked evaluation, async handoff, staging ingest/erasure, DPO sign, and L8 supplementary witness with ledger_record_verified: true.

cd E:\Documents\gnce\gnce-v2.4.0

.\scripts\run_at1_staging_real_l8.ps1

What it runs:

Step	Action
[1/3]	evaluate_intent on healthcare PHI scenario (Scenario 3 intent) → SARS on ledger → enqueue AT-1 handoff
[2/3]	Worker pass 1 — POST /v1/archives → 201, ERASURE_PENDING
[3/3]	Worker pass 2 — POST /v1/archives/{ref}/erasures → DPO sign → L8 seal (same GNCE_LEDGER_DIR)

Optional flags:

Flag	Effect
-NoPause	Skip "Press Enter to close" (use in an open terminal)
-NoL8Seal	Erasure + sign only; skip L8 (not recommended for full proof)

Pass markers (pass 2 JSON):

- archive_unchanged: true
- signature_valid: true
- **ledger_record_verified: true**
- erasure_status: ERASURE_WITNESSED
- pending_remaining: 0

Local artefacts (not for publication): per-run ledger and queue under .gnce/at1_real_l8/ (timestamped ledger_* and queue_* directories).

7.4.4 How staging relates to the bundle

	Bundle (run_all.ps1)	Staging smoke	Staging real L8
AT-1 surface	Local CLI archive	Hosted HTTP API	Hosted HTTP API
Evaluation	Pre-generated ADRA fixtures	None (synthetic HTTP payloads)	Real evaluate_intent
L8 seal	Yes (bundle witnesses)	No	Yes (ledger_record_verified: true)
Cite in paper	manifest.json hashes	Connectivity proof only	Part IV row 5; run-specific IDs

Recommended order: bundle first (reproducible CI proof) → staging smoke (API up) → run_at1_staging_real_l8.ps1 (full chain). See also Part IV row 5 and 3.7.

7.5 Regression gate

```
py -m pytest tests/ -q
```

Expected: 722/722 at time of writing (August 2026 estate) — GNCE automated regression suite only (includes L8 erasure witness, data persistence contract, and scenario 4 bundle test). This

count does not include an AT-1-side conformance suite; AT-1 verification is covered separately in 3.7 and the bundle/staging artefacts.

Appendix A — File Index

Repository root: gnce-v2.4.0 (GNCE Labs). Paths are relative unless noted. **Do not commit secrets** — staging credentials and DPO private keys stay local.

A.1 Synthetic integration bundle (reproducible proof — cite in paper)

Path	Description
integrations/at1_gnce_bundle/manifest.json	Scenario summary, payload hashes, pinned @tinyfiles/cli@0.1.81
integrations/at1_gnce_bundle/scenarios/*.at1.json	AT-1 input arrays (flat records + payload_hash binding)
integrations/at1_gnce_bundle/adra_receipts/*_adra.json	GNCE ADRA manifests per scenario
integrations/at1_gnce_bundle/certificates/*_erasure_cert.json	GNCE DPO–signed AT-1 erasure certificates
integrations/at1_gnce_bundle/witnesses/*_l8_witness.json	GNCE L8 supplementary erasure witnesses
integrations/at1_gnce_bundle/archives/<scenario>/	Local AT-1 archive state (vault, archive.json, erasure log) from bundle CLI runs
integrations/at1_gnce_bundle/run_all.ps1	End-to-end bundle runner (scenarios 1–4)
integrations/at1_gnce_bundle/l8_chain_verification_export.json	Constitutional hash-chain excerpt + verification metadata for independent auditors

integrations/at1_gnce_bundle/dylan_handoff_manifest.json	Per-file SHA-256 index for author review handoff
integrations/at1_gnce_bundle/dylan_handoff_bundle.zip	Packaged auditable artefacts (no secrets)

A.2 Production handoff — GNCE modules (schema v1.2)

Path	Description
gnce/integrations/at1_handoff.py	Post-eval envelope builders, persistence routing, evaluate_and_enqueue_at1_handoff
gnce/integrations/at1_handoff_queue.py	File queue (pending / completed / DLQ), DSAR partial alerts, archive index
gnce/integrations/at1_handoff_worker.py	AT-1 HTTP client, ingest/erasure delivery, DPO sign, L8 seal
gnce/integrations/at1_cert_signing.py	Ed25519 signing over AT-1 unsigned_certificate_bytes (wire-exact)
gnce/integrations/at1_archive_index.py	sars_id / subject_id → archive_ref index for DSAR fan-out
gnce/gn_kernel/phases/l8_erasure_witness.py	L8 supplementary witness build, verify, and hash-chain seal
gnce/gn_kernel/phases/data_persistence_contract.py	Persistence scope and erasure status contract
docs/AT1_HANDOFF_SCHEMA.md	Canonical handoff API contract (v1.2) — shared with AT-1 author

A.3 Staging validation (hosted AT-1 — August 2026)

Path	Description
staging/GNCE_staging_credentials.txt	Staging base URL + bearer token (local only; gitignored)
staging/README.md	Staging credentials placement notes
scripts/smoke_at1_staging.ps1	Staging smoke: health → version → ingest → erasure → DPO sign
scripts/smoke_at1_staging.py	Python runner invoked by smoke PowerShell wrapper
scripts/run_at1_staging_real_l8.ps1	**Full proof:** real evaluate_intent → staging ingest/erasure → L8 with ledger_record_verified: true
scripts/run_at1_staging_real_l8.py	Step 1 helper: evaluate + enqueue handoff (healthcare PHI scenario)
scripts/run_at1_staging_loop.ps1	Synthetic sars_id staging loop (ingest → erasure → sign; use -NoL8Seal if no ledger row)
.gnce/at1_real_l8/	Per-run ledger + queue artefacts from staging L8 runs (local; not for publication)

A.4 Local pilot and mock AT-1

Path	Description
scripts/at1_handoff_mock_server.py	In-process mock AT-1 (POST /v1/archives, /erasures) for offline demos
scripts/demo_at1_handoff_pilot.ps1	Full local pipeline: enqueue → worker ingest → erasure → sign → L8
scripts/run_at1_handoff_worker.py	Worker CLI entrypoint (--once, --queue-dir, --no-l8-seal)
gnce-dpo.key / gnce-dpo.key.pub	GNCE DPO Ed25519 key pair (repo root; **never commit private key**)

A.5 Verification, generation, and audit tools

Path	Description
scripts/verify_l8_witness_bundle.py	Independent cert + L8 witness verification for auditors
scripts/verify_l8_chain_export.py	Verify L8 witnesses on constitutional hash chain; regenerate export JSON
scripts/seal_at1_erasure_witness.py	Seal bundle erasure certificates as L8 witnesses (--cert, --adra-manifest, --all)
scripts/generate_at1_gnce_bundle.py	Regenerate bundle scenarios, ADRA receipts, and AT-1 inputs from sovereign tests
scripts/generate_at1_dylan_handoff.py	SHA-256 manifest + ZIP handoff package for AT-1 author review
tests/test_at1_handoff.py	Automated regression gate for queue, worker, signing, and L8 handoff path

crypto_shred *(GNCE) and* key-only *(AT-1) are equivalent terms for cryptographic erasure via key destruction.*

A.6 Operational runbooks

Path	Description
docs/FRIDAY_1500_AT1_HANDOFF_RUNBOOK.md	Live demo runbook (mock + schema v1.2 talking points)
docs/joint_writeup_gnce_at1_two_halves_one_audit.md	In-repo markdown source of this joint paper

A.7 Citation guidance

Artefact type	What to cite
Bundle scenarios 1–4	Fixed paths under integrations/at1_gnce_bundle/; pinned CLI 0.1.81
Staging runs	Run-specific sars_id, archive_ref, payload_hash, and L8 witness JSON under .gnce/at1_real_l8/ — hashes differ each run

Chain lookup	constitutional_hash / integrity_hash, not chain position alone

Appendix B — Suggested Next Work (Joint)

Item	Status
Production handoff API (queue + worker v1.2)	**Done**
Staging cutover (GNCE_AT1_BASE_URL)	**Done** (August 2026)
Regulator export bundle (single ZIP)	Still open
Dual-sign SLH-DSA path	Done
Sector pilots (healthcare)	Not started

The standalone verifier (scripts/at1_erasure_cert_verifier.py) validates the Ed25519 member only; dual-signed (/2.0) envelopes report UNSUPPORTED (exit 3) for the SLH-DSA member by design.

Appendix C — Author Notes for Publication

Publication tatus. Parts I–VII and appendices form a combined draft. Part II (GNCE) and Part III (AT-1) are complete and cross-verified in both directions (August 2026). Integration bundle scenarios 1–4 remain the pinned reproducible proof (@tinyfiles/cli@0.1.81). Staging runs (August 2026) demonstrate production-shaped async path (staging); cite run-specific sars_id, archive_ref, and L8 witness JSON rather than bundle fixture hashes. — safe for public citation**.**

Audit Reference Practice. Do not cite specific chain positions as permanent — they increment on each seal and may collide on legacy chains. Cite payload_hash equality across receipt, certificate, and L8 witness, and archive_hash_before == archive_hash_after. Prefer integrity_hash / constitutional_hash as the auditor anchor for chain lookup**.**

Staging validation (12 August 2026). sars-499156122bb046a7; archive_ref arc-a579383f2bed8f0f; issuer_key 8cf284cc...; full certificate and witness in .gnce/at1_real_l8/queue_20260812_183052/.

Reference Citation. *Nsukuzonke, V. & Wolpe, D. (2026). *Two Halves of One Audit: Constitutional **AI** Governance Meets Cryptographic Erasure. * GNCE Labs × AT-1 integration reference.*

Intellectual Property and Technology Ownership

GNCE Labs. GNCE is based on a patent-pending governance architecture associated with South African patent application ZA 2025/10023. The GNCE architecture described in this paper includes deterministic policy enforcement, constrained rationale synthesis, cryptographic execution gating, and sovereign audit/evidence mechanisms. GNCE retains all right, title and interest in the GNCE Technology and the intellectual property rights therein.

AT-1 / TinyFiles. AT-1 provides proprietary technology for verified-lossless compression, tamper-evident archives, cryptographic erasure, and independently verifiable erasure certificates. AT-1 retains all right, title and interest in the AT-1 Technology and the intellectual property rights therein. The AT-1 Technology is incorporated into the integration described in this paper under the collaboration and technology licence arrangement between the parties.

Joint Integration. This paper documents the technical integration between GNCE and AT-1, including the interfaces, handoff contracts, cryptographic bindings, verification procedures, and jointly validated evidence chain. The integration does not transfer ownership of either party's pre-existing technology or intellectual property. Ownership and licensing of any jointly created material or other integration-specific developments are governed by the parties' Collaboration and Technology Licence Agreement.

Nothing in this paper is intended to amend, replace, expand, or otherwise alter the intellectual-property ownership or licensing rights established by that agreement.

*End of Whitepaper— both halves complete. *