

Verification and measured evidence



Technical annex for engineering and assurance teams

Record checks / source provenance / evaluation results

September 2026 | Nage-reported evidence

Independent record checking

A reference verifier checks a specific exported record within an explicit disclosure and trust boundary.

Reported cross-language test

Python and TypeScript verifiers reached the same verdict on six checks for the same sample record. Each rejected three deliberately corrupted cases. These are interoperability and mutation observations, not exhaustive security coverage or an external audit.

Record integrity	Required inputs Signed body, declared canonicalisation version and trusted production public key.
Internal consistency	Checks Declared arithmetic, required null reasons and manifest structure or consistency.
Claim-support recomputation	Conditional Claim and evidence text, method identity and any required implementation artefacts.
Coherence and attention	Unavailable from record alone Embedding or worker-state dependencies prevent standalone recomputation in the current package.
Answer truth or execution	Outside this check Neither a valid signature nor log inclusion establishes the factual truth of the answer or proves the declared inference ran.

The distributed sample uses a deterministic test key. A real audit requires a separately authenticated production key and a real exported record.

A reproducible verification session

The demonstration should distinguish sample data from production evidence.

1 / Obtain the right artefacts

The external package includes verification/ with two reference verifiers, an English synthetic sample, a Unicode regression fixture and test public keys. For a production assessment, obtain a real disclosed record and authenticate the issuer's production key separately.

2 / Run the reference checks

From verification/, run: `python3 verify.py sample/receipt.json --pubkey sample/pubkey.txt --self-test`. Python 3.9+ is required; Ed25519 needs cryptography. Node 24 can run `verify.ts` directly using built-in cryptography. Exact setup and commands are in the English README.

3 / Review each verdict

A pass should name the check and its input scope. Missing dependencies or undisclosed evidence must stay unmeasured. Do not collapse partial coverage into a general verified label.

4 / Test rejection

Change signed content, alter the signature and violate the declared confidence-gap relationship. Rejection shows that those mutations are detected. It does not show resistance to every attack or verify the semantics of the source claims.

5 / Retain the result

Keep the record, key identifier, verifier version, dependencies and output together. The included evidence directory contains the actual local demo run. These checks assess specific record properties; they do not replace a complete conformance assessment.

English output is included in verification/evidence/. Commands run from the accompanying verification/ folder. The fixtures are synthetic test records.

The included demonstration

Two fixtures, two reference implementations, with observed output retained.

English fixture

Synthetic / test key

An English demonstration generated with an explicitly public deterministic test seed. It is not a production inference.

Unicode fixture

Original supplied sample

Preserved unchanged to exercise non-ASCII text. The associated key is also a test key.

Record checks

6/6 PASS on each fixture

Both Python and TypeScript passed the six listed checks for each fixture in the local run on 15 September 2026.

Mutation checks

3/3 detected on each fixture

Both implementations rejected the three deliberate corruption classes in their self-tests. This is not exhaustive security testing.

Output comparison

Matching human-readable output

Verification and self-test text matched exactly across implementations. Python supplies --json; the supplied TypeScript CLI renders text.

Where to inspect the evidence

See verification/evidence/ for outputs and CHECK_SUMMARY.json for runtime versions. The verifiers are copied from the supplied repository; fixture generation is separately identified.

These are local synthetic-fixture observations, not a new production benchmark or an independent organisational audit. Sample keys do not authenticate Nage production records.

Canonicalisation and trust

Independent implementations need the same signed bytes, not merely the same-looking JSON.

Versioned byte representation

The updated source register reports all three defects closed in the signed-record path, using sha256-canonical/2. A separate request-payload hash in audit.py still uses raw UTF-8. That remaining code difference does not alter the signed body checked by the reference verifier.

Interoperability requirements

Signed-record and request-hash formats have distinct scopes. Keep their algorithm identities and historical compatibility explicit. Release-matched vectors should test Unicode escapes, numbers and empty metadata. Request-hash unification remains separate work requiring migration planning.

Signing and key trust

Signature verification binds a body to a private key. The reviewer separately needs confidence that the corresponding public key belongs to the intended issuer. Rotation, revocation and historical verification policy belong in the operational contract.

Chain and execution limits

The chain uses HMAC. The Merkle root remains unsigned and unpublished; sign and then publish it before describing it as a public commitment. Inclusion proofs are reported as implemented but unpublished. Inclusion alone would establish membership, not execution or complete omission detection.

Reference: RFC 6962, audit paths and consistency proofs. Use the exact verifier release and canonicalisation version declared by the record.

Source hashing and serving operations

Reported observations should retain their unit, origin and boundary.

7,230 chunks	Zero reported hash mismatches Backfilled hashes matched a database digest. They identify text held at backfill time, not uninterrupted historical integrity.
Three execution environments	Matching digest outputs Mac and Railway used hashlib; PostgreSQL used pgcrypto. This is a cross-environment check, not three independently written algorithms.
7,230-leaf Merkle root	Matching Python / SQL folds The root remained unsigned and unpublished in the supplied report. No public inclusion commitment is asserted.
Historical source chain	Coverage incomplete Migration 043 addresses future rows. Historical source and extraction hashes remain incomplete; backfilling them requires rereading original bytes from object storage.
Serving-path deletion	Removal / zeroing reported Durable artefacts in object storage remain outside the same delete operation. Base-model and overlapping-source knowledge are separate concerns.
186 s to 23.3 s	Reported retrieval operation The reported operation became approximately 8 times faster with identical scores. This is not an end-to-end latency SLA or a quality gain.

Sources: measured-results annex B.2, B.3 and B.7. Production readiness and performance require checks on the intended deployment.

Claim support and contradiction

The negative result is material to the meaning of the signal.

Constructed evaluation

The development set contained 105 items: 29 supported, 64 contradicted and 12 silent. The live lexical method scored contradicted claims at a mean of 0.890 and supported claims at 0.780. The reported K3 AUC was 0.008 against a predeclared 0.65 threshold.

Separate contradiction signal

A deterministic detector uses numeric, date, polarity and entity checks with a topic-overlap gate. The team tuned it on development data and evaluated a holdout of 95 items.

Contradiction recall

88.1%

Reported holdout detection of contradicted claims.

False positives on support

11.5%

Supported claims incorrectly flagged as contradicted.

False positives on silence

0.0%

Observed only in the silent subgroup. It is not the overall false-positive rate.

Deployment status

The new detector is test-stage and not yet connected to the live gamma signal. Random production sampling has not been established. The constructed set and document clustering limit generalisation; reported intervals should not be treated as deployment-wide guarantees.

Source: claim-support technical talk and measured-results annex B.3. A high overlap score without a contradiction flag is not sufficient proof of support.

Craton experiments and measurement repair

Evidence delivery must work before an evidence-use experiment can be interpreted.

What changes next

The evidence-use comparison needs to be rerun after the packing repair. The reported fourth-stage reading-loss measurement has not run. Legacy left truncation recorded zero loss in this run; it was a latent risk after increasing the earlier character budget, not an observed contributor to all of the measured loss.

Capacity comparison	252M to 642M Reported A score changed from 8.0% to 8.8%, described as within noise. This is one experimental setting.
Weight path	0/15 held-out The route-to-one test did not demonstrate useful performance in the reported setup.
Sterility probes	10/10 A limited test observation, not proof that the base contains no factual information.
Retrieval	221/250 items The router found evidence in 88.4% of the tested items.
Legacy packing	103/221 retained 118 retrieved items lost visible evidence before the model, a 53.4% loss among retrieved items.
Repaired packing	221/221 visible At the reported 8,000-token budget, all retrieved evidence was visible. This measures prompt visibility, not answer correctness.

Source: sealed-prediction methodology and measured-results annex B.4. The test uses 250 items from one constructed world.

Evidence register and next checks

A usable diligence package links every claim to a versioned artefact.

Verification release

The accompanying verification/ folder includes the two reference verifiers, English instructions, test fixtures and local run evidence. Signed-record fixes are reported closed. Request-hash migration and production-key authentication remain separate requirements.

Production evidence

Provide exported records at the agreed disclosure level, a capability-specific readiness statement and an ingestion-versus-backfill coverage report. Clearly identify historical gaps.

Evaluation artefacts

Provide the dataset construction protocol, development and holdout assignments, metric definitions and run outputs. Add production-distribution evaluation before selecting operational thresholds.

Craton acceptance

Repeat mechanism tests after repairing evidence delivery. Define the intended sterility domain and evaluate utility, isolation and evidence use together.

Source register

Nage conference package, 14 September 2026: Independent Verification Annex; Measured Results Annex; Claim-support K3 technical talk; Sealed Predictions methodology; Provenance slide; KLM Specification.

External protocol reference: <https://www.rfc-editor.org/rfc/rfc6962>

This annex selects verification and diagnostic results. The source register separately permits SEDIM-Bench 90.6 with its nage-8b v3.1 / table2.tex scope; the underlying artefact was not reproduced here.