How a machine checks a stored industrial claim

Question: can the factory rely on “battery 42 has class B”?

A Graph proof in the evidence store

1 Retrieve the signed snapshot
Get its original statements, proof options and signature from the evidence store.

2 Rebuild and check the proof
Put statements in the agreed standard order (canonicalisation), hash them and verify the issuer’s signature and proof options.

The graph path uses the retained signed dataset. The nested path reconstructs a graph view from authenticated source claims.

3 Match the fact being used
Check that “battery 42 has class B” belongs to this verified source snapshot.

B1 JSON-LD in SD-JWT (envelope only)

1 Retrieve the original token
Also retrieve the disclosed values and salts needed for the claim.

2 Unpack and verify the envelope
Read the header and claims; check the outer signature. Decoding is not verification.

3 Reconstruct the disclosed claims
Check each supplied disclosure digest. Restore values only where they are allowed.

4 Turn claims into graph statements
Apply the approved vocabulary, context and mapping; keep the source attribution.

5 Bind the graph fact to its source
Check that the stored fact matches the verified claims under the approved mapping.

BOTH PATHS Check authority, status, freshness and the decision policy
Use only the verified view. Reject or hold the decision if a required check fails.

Measured prototype portions — one 146-quad credential

Graph retrieval and proof check: 0.28 ms

Envelope processing and fresh graph expansion: 1.06 ms

About 3.8× per additional check for those timed paths; not a whole-workflow comparison.
The envelope-to-cached-graph comparison, production trust and network costs are excluded.

A cached envelope-only check took 0.25 ms, excluding checks of its mapping and graph binding.
Graph proofs cost more initially; in this fixture they become cheaper than fresh expansion from the fifth additional check. A cache can avoid re-expansion only with a governed binding.