WHITE PAPER · SAOLIX EAGLE · DRAFT 1 · 20 OCTOBER 2026

The determinism boundary, as built.

What a regulator will ask about an AI-assisted screening system is not what the model can do, but what it cannot. This paper describes, from the Saolix Eagle code base and its architecture decision records, how the verdict is kept deterministic, how its record is sealed and verified, how names are matched across alphabets, and, in its own section, what is not built yet.

Statements here describe Saolix Eagle as of the date above. Features change; confirm in writing anything you rely on.

1 · THE LINE

AI assists. Rules and people decide.

Eagle draws one line through its pipeline. Above the line, software may be probabilistic: a name-matching engine scores candidates, and a local language model may re-rank an alert queue and draft a plain-language explanation. Below the line, the verdict on each screening comes from a pinned, versioned ruleset, or from a person. Nothing a model produces is a decision.

The rule is recorded as an architecture decision, "AI assists, never decides", and it is enforced in four places rather than relied on as a convention:

A build guard

A check in the build pipeline fails the build if any code path lets a model output become a verdict.

A database constraint

The decisions table accepts a decision source of RULE or HUMAN only. A row that claims a model decided cannot be written.

A runtime assertion

Every decision is checked at the moment it is recorded; a source other than a rule or a person stops the write.

Triage guards

If a model labels a true list hit as noise, the guard returns the case to review. A model signal can raise a result; it cannot lower one.

The language model itself is a small open model run locally on the customer's hardware, with its weights pinned by a SHA-256 digest, behind a single port in the code so that no other path to a model exists. It sees the candidates and the reasons; it never sees a button that closes a case.

2 · SAME INPUT, SAME OUTPUT

What makes a verdict reproducible.

A verdict is reproducible only if nothing in its path depends on the moment it ran. Eagle's verdict core is kept free of the usual sources of drift by a build guard that fails the build if the core reads the clock, draws a random number, sorts by locale, uses locale-aware formatting, or reads the environment. Wall-clock time is read in exactly one place, outside the core, and passed in.

Every screening is stamped with the version of the sanctions list it ran against and the version of the rules it ran under, so a result can always be tied to the exact inputs that

produced it. Changes to the governed configuration at runtime, such as a threshold, require a written rationale and a named approver, and each configuration version is stored with a snapshot of the engine and chained by hash to the one before it.

The scoring is fixed and published inside the product: three signals, a character n-gram similarity, a fuzzy string similarity and a full-text match, fused with fixed weights; a clear result below 0.65, review between 0.65 and 0.85, and a fail at 0.85 and above; an exact lexical match floors the score at 0.95; and a candidate whose tokens are fully contained in the subject's name is forced to review whatever its score. There is no neural embedding in the verdict path: the "semantic" signal is a deterministic hashing of character n-grams.

Byte-for-byte reproducibility is proven in the build: a fixed set of test vectors is normalised and scored on every build and the digest of the output is compared with a committed golden value. A changed byte fails the build.

3 · THE RECORD

How a screening is sealed, and how tampering shows.

Each interactive screening writes an audit entry together with its decision: a sequence number, the name searched, the result count, the acting user, the list version, the tenant, the outcome, the request identifier, the details and the timestamp. The entry is serialised in a fixed field order with sorted keys, so the same entry always produces the same bytes, and sealed.

The live system of record is a Merkle log in the style of RFC 6962, the structure used by certificate transparency: every entry is a leaf, the tree head commits to every entry so far, and the log can produce an inclusion proof for any single entry and a consistency proof between any two tree heads. A linear hash chain, each entry hashed with the one before it, is kept as a fallback. Seals are keyed with a secret held in the deployment environment and never in the database, so a party with database access alone cannot re-seal what they change.

The audit schema is written by a database role that can only insert and read, never update or delete, and write-once triggers back that up. An integrity check, available on demand through the product's interface for the whole log, for the general log or for a single request, verifies the seals and returns the first sequence number at which anything is broken. When personal data reaches the end of its retention period, the entry's personal fields are blanked while its hash skeleton is kept, so the log still verifies after disposal.

4 · NAMES

Matching across alphabets.

A sanctioned name can arrive in Cyrillic, Arabic, Chinese, Korean or Japanese script, in full-width characters, with diacritics, misspelled, or disguised with look-alike characters from another alphabet. Before any scoring, Eagle normalises both sides: Unicode compatibility decomposition, transliteration of Cyrillic and Arabic to Latin, romanisation of Chinese, Korean and Japanese, folding of full-width forms, stripping of diacritics, folding of homoglyphs, and a fixed ordering of tokens. The comparison then draws on Jaro-Winkler, Double Metaphone and Daitch-Mokotoff phonetic encodings, trigram full-text matching and the n-gram similarity described above. All of it is deterministic.

5 · WHERE IT RUNS

Inside your perimeter.

Eagle ships with an air-gapped deployment in which the database, the application and the model gateway share an internal network with no route out, under a sovereign mode that assumes no outbound connection exists. A build guard keeps the verdict path free of outbound calls and of GPU dependence, so the product runs on ordinary CPU servers inside a bank's own data centre. The local model is a prerequisite only for AI assist: with it absent, screening runs unchanged and the assist features are simply off.

6 · NOT YET BUILT

What is planned.

A paper like this is only useful if it also says what is not there. These are recorded as open items in Eagle's own decision records, and will move into the sections above when they are built.

- A regulator-replay endpoint that re-runs a stored decision from its own audit record and returns a byte-identical result. Today, byte-for-byte replay is proven in the build against pinned test vectors, and every screening carries the ruleset and list versions it ran under.
- A Saolix-owned inference runtime for the local model. Today the model runs on the customer’s hardware through Ollama, a third-party prerequisite; without it, AI assist is off and screening is unaffected.
- A public-key signature or external anchor on the audit tree head. Today the tree head is keyed with a secret held outside the database, so only a holder of that secret can verify it.
- Per-screening pinning of the governed configuration version, alongside the ruleset and list versions already pinned.
- A prompt registry, so that the exact prompt a model saw is versioned with the decision.
- Guards, not only tests, for two rules: that a model signal can only raise a result, and that switching a feature off gives byte-identical output.

7 · SEE IT

Ask us to show you, not tell you.

Everything in sections 1 to 5 can be shown in a working session: the build guards failing on a planted violation, the integrity check run on a live log, and the replay conformance test. Further reading from the group: **The determinism boundary in practice** ↗ and **Tamper-evident audit: how Eagle seals a screening** ↗.

Bring your model-risk team.

We will walk through this paper against the running product, with your questions.