The Autonomous Execution Proof: Structure and Verification Model

Intentrax Labs Research Paper | Execution Assurance
Published 2026-07-09 - Intentrax Labs - www.intentrax.com/labs

How an execution event, a bound policy and authority, and a deterministic decision are composed into a single tamper-evident, replay-verifiable artifact.

Role of the artifact

The Autonomous Execution Proof (AEP) is the canonical cryptographic artifact produced by the Intentrax Assurance Engine under AEVN. It is a tamper-evident, replay-verifiable, audit-ready evidence artifact of autonomous execution. An AEP is engine-generated: the executor cannot edit it after the fact. From a single AEP an independent verifier can determine what execution evidence was normalized, which principal and authority were bound, which policy governed the event, which decision was produced, how integrity and replay references were preserved, whether the proof was altered, and whether the evidence can be checked outside the runtime. The AEP is not a log summary and not an observability trace; it is the load-bearing evidence object.

Inputs: EEO, POD, and the decision object

The AEP is composed from the protocol-layer artifacts AEVN defines. The Execution Event Object (EEO) is the normalized record of what was executed, carrying its own event_version. The Policy Object Definition (POD) is the policy snapshot that governed the event, carrying its own policy_version and referenced by policy_id and policy_hash. Before proof generation, the engine produces a deterministic Decision Object recording the result of policy evaluation: the decision, a decision_reason_code, the policy identity and hash, and the evaluation reference. Every canonical artifact also carries the protocol field aevn_protocol_version, which fixes which verification rules interpret it. The decision object is a required input to proof generation, which is what makes the decision itself replayable rather than merely asserted.

The dual-gate, fail-closed decision

The decision bound into an AEP is produced by a dual-gate model. An ALLOW is emitted only if both the execution gate and the authority gate permit the action; otherwise the outcome is DENY. This makes authority binding first-class rather than advisory: the proof records not only that an action occurred but that it occurred under an explicit policy version and a specific identity, delegation, and constraint set. The system is fail-closed by construction. If deterministic validation cannot be completed, trust is not granted: verification fails, no certificate is issued, an execution passport is not trusted, execution is not approved, a failure code is returned, and an audit event is recorded. The cost of this design is more DENY outcomes; the benefit is that a permissive result is never the default when evidence is

incomplete.

Integrity and portability

The AEP's integrity rests on canonicalization followed by hashing. Canonical protocol data is serialized under deterministic rules and hashed with SHA-256 (MVP), producing a proof hash that any independent party can recompute. Hashes carry an explicit hash_algorithm identifier and signatures carry signature_algorithm and key_id fields, so the protocol supports future algorithm and key-rotation upgrades without breaking existing proofs. Operational metadata such as request identifiers, traces, and network details may be stored or displayed but never participates in canonical hashing. The signed AEP is portable: it can be exported as JSON, aep-json, or a human-readable PDF companion for audits, disputes, and claims, without the recipient needing access to the originating runtime.

Relationship to receipts, passports, and certificates

The AEP has a fixed position among adjacent artifacts, and the boundaries are strict. An Action Receipt is a developer-facing, non-authoritative local summary; it may link to AEP exports and registry references but never replaces canonical AEP proof, and it explicitly marks itself as non-authoritative. An execution passport is ingested external evidence, bound to but never replacing the AEP. A certificate is the signed AEP presented as an attestation. Full AEP JSON export remains the authoritative verification material after runtime admission; presentation artifacts derived from it are non-authoritative. No workspace, console, or workflow layer may modify an AEP, its proof hashes, its verification results, or its signature bundles outside the deterministic system that owns them.

Authority boundary of this document

This explanation is downstream of the locked contracts and verifier vectors and does not define them. It must not publish permissive placeholder schemas or redefine the proof locks; the machine-readable schemas and golden vectors remain in the contracts corpus. The public conformance entry point verifies the hash-bound proof export fixture, the standalone verifier golden, and the public SDK receipt and export vectors while preserving this boundary. A note on planned scope: graded Assurance Levels 1 through 4 are planned and are not asserted by the present AEP, and a dedicated proof-generation runtime service remains deferred so that proof ownership stays inside the engine during beta rather than being split across services.