

SINT PROTOCOL

Runtime Authorization and Evidence Logging for LLM-Driven Physical AI Systems

Illia Pashkov

SINT AI Lab / PSHKV Inc. · Los Angeles, CA
i@pshkv.com · github.com/sint-ai/sint-protocol

ABSTRACT

Large language model (LLM) agents increasingly issue commands to tools, robots, drones, smart-home devices, and industrial systems. No widely adopted runtime authorization model currently covers the full LLM-to-actuator path with graduated human oversight, physical-constraint enforcement, and tamper-evident evidence trails.

We present **SINT Protocol**, a capability-based runtime authorization framework for LLM-driven physical AI. SINT interposes a single Policy Gateway between agent intent and actuator execution. Every request is normalized, validated against Ed25519-signed capability tokens, classified into one of four approval tiers, evaluated against physical constraints, and recorded in a SHA-256 hash-chained evidence ledger.

On an Apple M3 Pro with in-memory persistence, the gateway adds 5.1 ms steady-state p99 latency over 600 single-process iterations — within the 10 ms budget of a 100 Hz ROS 2 control loop. We discuss mapped coverage of OWASP Top 10 for Agentic Applications and implementation-support evidence for IEC 62443, EU AI Act, and NIST AI RMF obligations. We identify open limitations: no real-robot validation yet, heuristic drift thresholds, no mechanized formal verification, and operator-burden risks.

SINT does not replace model alignment, hardware emergency-stop controllers, or domain-rated safety standards.

KEYWORDS agentic AI security · physical AI · capability-based security · runtime authorization · OWASP ASI · EU AI Act Article 14 · NIST AI RMF · robot safety · Model Context Protocol · evidence ledger

Illia Pashkov — *SINT AI Lab / PSHKV Inc., Los Angeles, CA* i@pshkv.com — [git-hub.com/sint-ai/sint-protocol](https://github.com/sint-ai/sint-protocol)

Working paper — May 2026 — Apache-2.0 — Revision 1.1

Table of Contents

1. Introduction
 2. Background and Threat Model
 3. Design Principles and Architecture
 4. Core Primitives
 5. Safety Mechanisms
 6. Implementation
 7. Evaluation
 8. Open Problems
 9. Discussion and Limitations
 10. Related Work
 11. Conclusion
 - Acknowledgments, AI Disclosure, Funding, CRediT
 - References
 - Appendix A — Formal Invariants and DFA
 - Appendix B — Environment Variable Reference
 - Appendix C — Package Dependency Graph
 - Appendix D — Compliance Mapping Tables
 - Appendix E — Research Agenda 2026-2031 (*formerly §8; relocated for scope discipline*)
-

1. Introduction

1.1 The Physical-AI Transition

Foundation models have evolved on a steep trajectory of agency. Early LLM deployments treated the model as a text completion engine; the next wave added tool use, retrieval, and code execution; the current generation embeds models as policy heads for embodied agents. SayCan, Code-as-Policies, RT-X, and OpenVLA demonstrate end-to-end pipelines in which natural-language goals translate into ROS 2 topic publishes, MAVLink commands, or low-level motor torques [Ahn et al. 2022; Liang et al. 2023;

Open X-Embodiment 2024; Kim et al. 2024]. As of 2026, LLM-driven policy generation is being actively integrated into warehouse autonomous mobile robots (AMRs), collaborative industrial arms, BVLOS-capable delivery drones, and consumer smart-home agents. LLMs are also increasingly considered for high-consequence embodied workflows, including medical and industrial systems, although safety-critical certification (FDA Class III, IEC 62304 Class C) currently requires deterministic controllers below the LLM layer.

Three converging forces accelerate this transition. First, prompt-to-hardware tooling — iOrchestra, Siemens Industrial Copilot, ABB RobotStudio AI Assistant, Rockwell Emulate3D — has made it economical for non-specialists to *generate* industrial control plans from text. Second, the Model Context Protocol (MCP) [Anthropic 2024], Google A2A [Google 2025], and similar agent protocols have removed integration friction between LLM clients and downstream tools or actuators. Third, foundation models themselves have crossed thresholds at which deferring physical decisions to a human for every command becomes operationally untenable: a 100 Hz ROS 2 control loop running on a `/cmd_vel` channel has a 10 ms budget, while the median human approval latency is 200–500 ms — three orders of magnitude apart.

1.2 The Three Gaps

The acceleration documented above exposes three gaps. Existing systems address adjacent layers but do not combine runtime authorization, physical-constraint enforcement, and tamper-evident evidence across agent protocols.

G1 — The Authorization Gap. No widely adopted runtime authorization model currently covers the full LLM-to-actuator path. Current practice relies on (a) pre-deployment configuration files, which cannot react to dynamic context; (b) RBAC bolted onto agent runtimes, which conveys ambient authority and does not bind to physical envelopes; (c) ad-hoc allowlists embedded in bridges or tool servers, which scatter authorization logic across dozens of components without a common audit substrate. The Microsoft Agent Governance Toolkit (2026) provides runtime governance for *digital* agents but does not include physical-domain bridges, embedded physical-constraint primitives, or hash-chained evidence.

G2 — The Drift-Visibility Gap. Cardenas et al. [arXiv:2603.26997] measured up to a $4.8\times$ spread in out-of-policy action proposal rates across frontier LLMs on identical ROS 2 tasks, with a $3.4\times$ persistent divergence between best and worst backends. This is *not* an alignment failure — the models are instruction-following correctly — but a behavioral property that no amount of prompt engineering eliminates. Operators today have no standardized runtime metric to distinguish a cautious agent from a reckless one short of post-incident analysis.

G3 — The Static-Envelope Problem. Safety envelopes (maximum velocity, geofence polygon, maximum force) are inherently *static* at configuration time. The physical en-

environment is not: obstacles appear, humans enter workspaces, sensors degrade. A fixed-parameter envelope is therefore either too restrictive (blocking legitimate actions in benign conditions) or too permissive (allowing dangerous actions in adverse conditions). Closing this gap requires constraint tightening driven by real-time sensor state, which no current agent protocol expresses as a first-class primitive.

1.3 SINT Protocol — Position and Contributions

SINT Protocol addresses all three gaps by interposing a single runtime safety shield at the agent-actuator boundary. Within a deployment where all actuator paths are routed through SINT (see §3.5 *Deployment Trust Assumptions*), every action — whether an MCP tool call, a ROS 2 topic publish, a MAVLink command, an OPC-UA write, or a smart-home service invocation — flows through one `PolicyGateway.intercept()` choke point that enforces:

1. **Capability-token confinement** with embedded physical constraints (Ed25519-signed, attenuation-only, revocable in real time).
2. **Four-tier graduated human oversight** (`T0_OBSERVE` , `T1_PREPARE` , `T2_ACT` , `T3_COMMIT`) with consequence-based classification, contextual escalation, and optional M-of-N quorum on T3.
3. **Behavioral drift detection** via the Composite Safety Drift Score (CSDS) computed per `foundation_model_id` , that auto-escalates an agent’s tier when its measured behavior diverges from baseline (see §5.1 — formerly named CSDS in v1.0; renamed and corrected here).
4. **Environment-adaptive constraint tightening** via the `DynamicEnvelopePlugin` , which maps real-time sensor state (obstacle distance, human proximity) to tighter effective limits.
5. **Active threat-detection plugins** for prompt injection (`GoalHijackPlugin` , OWASP ASI01), memory poisoning (`MemoryIntegrityChecker` , ASI06), shell injection (`ArgInjectionDetector` , ASI05), and supply-chain tampering (`SupplyChainVerifier` , ASI04).
6. **Emergency stop** via `CircuitBreakerPlugin` , which supports implementation evidence for EU AI Act Article 14(4)(e)-style stop-control requirements, with three states (CLOSED → OPEN → HALF_OPEN), manual operator trip, automatic trip on N consecutive denials or anomalous drift, and probe-based recovery. SINT’s circuit breaker is a software-level control and complements, but does not replace, hardware-rated emergency stop systems under ISO 13849, IEC 61508, or domain-specific safety standards.

Every authorization decision is recorded in a SHA-256 hash-chained, append-only **Evidence Ledger** whose entries are independently verifiable via cryptographic proof receipts.

We make the following contributions:

C1. A complete protocol design centered on a *single choke point* with six formal invariants (I-T1/I-T2/I-T3 on tokens, I-G1/I-G2/I-G3 on the gateway DFA) that together establish bypass-impossibility for physical actuation *within a deployment that routes all actuator paths through SINT*. Invariants are enforced by runtime code in v1.0; mechanized formal verification is future work (§8 L1).

C2. A consequence-based four-tier classification system with consequence-aligned escalation triggers (`Δ_human` , `Δ_trust` , `Δ_env` , `Δ_novelty`), validated against six discipline-specific deployment profiles (warehouse AMR, welding arm, surgical Class III, drone BVLOS, collaborative robot, ROV).

C3. CSDS (Composite Safety Drift Score) — a sliding-window behavioral fingerprint metric per foundation model that operationalizes the ROSClaw 4.8× inter-model variance as a live enforcement trigger rather than a post-incident finding. The default weights are heuristic in v1 and require empirical calibration; they are not presented as universal constants.

C4. A reference implementation of 49 packages, 15 protocol bridges, three SDKs, and ~1,728 tests with mapped coverage of OWASP Agentic Security Top 10 (ASI01–ASI10) through conformance fixtures, running at sub-5 ms p99 latency on a 100 Hz ROS 2 budget *in single-process, in-memory benchmarks* (production-persistence and networked-gateway benchmarks are planned, see §7).

C5. A unified **compliance crosswalk** — providing implementation-supporting evidence, not certified compliance — mapping SINT mechanisms to IEC 62443 FR1–FR7, EU AI Act Articles 9/11/12/13/14(4)(e)/15, NIST AI RMF (MAP/MEASURE/MANAGE/GOVERN), ISO/IEC 42001, ISO 10218-1/2, ISO/TS 15066, IEC 62304, IEC 60601-1-8, FDA 21 CFR Part 820, ISO 14971, NIST SP 800-82, NATO STANAG 4586, ASTM F3548-21, FAA AC 107-2B, and DNV-ST-0111. Mapping is not certification; certified compliance requires accredited assessment.

C6. An explicit five-year research agenda (Appendix E) charting six open problems — swarm coordination, sub-human-reaction-time approval, model-bound capability tokens, IoT/edge authorization, side-channel hardening, autonomous economic coordination, and formal mechanization of gateway invariants — with planned publication targets and explicit key open questions.

1.4 Paper Organization

Section 2 lays out the threat model and the design space of existing agent and robot-safety protocols. Section 3 presents the protocol architecture, eight design principles, the request-lifecycle DFA, and the deployment trust assumptions on which SINT’s no-bypass property rests. Section 4 specifies the core primitives: `SintRequest` , `PolicyDecision` , the capability-token structure, the resource-URI scheme, and the four-tier system. Section 5 details the safety mechanisms: CSDS, dynamic envelopes, forbidden-combination detection, the five active plugins, and swarm collective constraints. Sec-

tion 6 describes the implementation: monorepo layout, the 15 bridge adapters, the engine layer, the evidence ledger, the economic layer, and the operator interface. Section 7 reports the evaluation: latency benchmarks (with reproducibility block), test inventory, OWASP ASI mapped coverage, six physical-AI deployment profiles, the compliance crosswalk, a comparison table against adjacent systems, and explicit *Evaluation Limitations*. Section 8 names the five open problems whose detailed agenda is in Appendix E. Section 9 discusses limitations and ethics. Section 10 surveys related work. Section 11 concludes. Appendix E contains the five-year research agenda (relocated from §8 in v1.0 for scope discipline).

2. Background and Threat Model

2.1 Empirical Foundations

We anchor SINT’s design in four empirical findings that, taken together, establish that physical AI cannot be secured by alignment alone.

E1 — ROSClaw inter-model variance (Cardenas et al., arXiv:2603.26997, 2026) [1]. Evaluated GPT-4o, Claude 3.5 Sonnet, Gemini 1.5 Pro, and an open-source baseline driving identical ROS 2 robotic tasks under matched system prompts. The study reports a **4.8× spread** in out-of-policy action proposal rate (highest divided by lowest backend) and a **3.4× persistent divergence** between the best and worst frontier backends. The variance is reproducible under repeated trials and survives prompt re-engineering. The implication is structural: a robot certified safe under one foundation model is not certified safe under any other foundation model. SINT’s CSDS metric operationalizes this finding by making the variance enforcement-visible at runtime.

E2 — MCP protocol-level vulnerabilities (arXiv:2601.17549, 2026). A formal security analysis of the Model Context Protocol identified three protocol-level vulnerabilities and constructed 847 attack scenarios spanning tool poisoning, prompt injection via tool descriptions, command injection, cross-server escalation, server impersonation, and supply-chain tampering of tool servers. The same study reports attack amplification of 23-41% relative to baseline LLM tool-use without MCP, and demonstrates that a proposed authorization layer (MCPSec) reduces attack success from 52.8% to 12.4% at an 8.3 ms enforcement-path overhead. MCP itself contains no authorization framework — it standardizes the *protocol* for tool calls but delegates all access decisions to the host application. SINT addresses ASI categories 1, 3, 4, 5, 6, 7, and 10 that are directly inherited from the MCP attack surface, and the 12.4% post-mitigation residual rate establishes the baseline against which SINT’s plugin pipeline (§5) must be measured.

E3 — SROS2 critical vulnerabilities (CCS 2022). Formal analysis of SROS2 — the security extension to ROS 2 — uncovered four critical access-control vulnerabilities

permitting arbitrary command injection and key exfiltration. The failure mode was not implementation bugs but the underlying architectural choice to ship policies as static XML files loaded at node startup, with no runtime enforcement choke point. SINT’s “single gate, hot-revocable” architecture is a direct response to this finding.

E4 — Consumer-robot firmware vulnerabilities (2025). Public disclosures during 2025 included reports of consumer humanoid robots shipping with hardcoded BLE/Wi-Fi cryptographic keys, raising the possibility of wormable command injection across fleets. We cite this class of failure illustratively rather than as a verified incident: the broader point is that device-level authentication is consistently weaker than vendors claim, and that per-agent capability tokens with real-time revocation — exactly what SINT provides as a layer *above* device authentication — would enable instant containment regardless of whether the device’s own credentials are compromised.

2.2 Why Existing Protocols Do Not Solve This

Existing agent and robot protocols address overlapping but distinct concerns. Table 1 (§7) presents the full comparison; here we identify the structural gaps.

- **MCP** [Anthropic 2024] standardizes tool-call exchanges between LLM clients (Claude Desktop, Cursor, GPT clients) and tool servers. It has no authorization framework, no physical-constraint primitives, no graduated oversight, no audit trail.
- **A2A** [Google 2025] standardizes inter-agent communication. It conveys task delegation but contains no capability or constraint primitive and no audit substrate.
- **ACP** (IBM/Linux Foundation) provides agent-communication conventions with partial graduated approvals but no physical-domain primitives.
- **AgentProtocol**, **ANP** (Agent Networking), and **AG-UI** target API surface uniformity, networking, and UX streaming respectively — none address authorization, physical constraints, or audit.
- **SROS2** secures ROS 2 inter-node traffic via DDS but does not bind authorization to physical envelopes and ships static policies (as documented in [CCS 2022]).
- **Microsoft Agent Governance Toolkit (2026)** addresses OWASP ASI for *digital* agents (software-only); it does not provide robot/drone/industrial bridges, physical-constraint enforcement, or hash-chained evidence.

SINT is not a competing agent protocol. It is a **security enforcement layer** that sits between any agent protocol and the physical world. MCP + SINT = secured MCP. A2A + SINT = secured A2A. ROS 2 + SINT = secured ROS 2.

2.3 Threat Model

We consider an LLM-driven agent that controls physical actuators or digital tools via one of 15 bridge adapters (MCP tool calls, ROS 2 topics/services/actions, MAVLink commands, OPC-UA node operations, MQTT/Sparkplug messages, Open-RMF fleet tasks, A2A inter-agent calls, gRPC services, Home Assistant services, Matter commands, FHIR resource access, swarm coordination, IoT MQTT/CoAP, economic operations).

Trust boundaries (per `docs/security/formal-threat-model.md`):

1. Agent runtime → bridge adapters
2. Bridge adapters → policy gateway
3. Policy gateway → approval system / operators
4. Policy gateway → physical executors (robots, PLCs, drones, IoT, smart-home, health devices)
5. Gateway → evidence ledger and persistence backends

Adversary model. The agent is **authenticated but untrusted**. It may:

- Hallucinate dangerous commands.
- Be prompt-injected (ASI01) — including by content embedded in tool parameters, tool descriptions (ASI03), or response data from other tools (ASI07).
- Drift behaviorally across conversation turns (ASI10).
- Have its tool chain or model fingerprint tampered with (ASI04).
- Attempt to read credentials or PII (ASI06).
- Be coerced into shell or code execution (ASI05).
- Attempt to escalate via misuse (ASI02).
- Trigger cascading failure via storm-of-denials (ASI08).
- Exploit delegation trust (ASI09).

Operator trust. The human operator is **trusted but fallible** — possibly absent, distracted, or overwhelmed by alert volume. The protocol must not assume continuous human availability for sub-second decisions.

Cryptographic assumptions. Ed25519 signature unforgeability holds; SHA-256 collision resistance holds; operators enforce private-key custody and infrastructure access controls; physical safety controllers (hardware E-stop, ISO 13849 PL e light curtains) remain the final interlock layer below SINT.

Out of scope. Side-channel attacks on the cryptographic primitives themselves; nation-state hardware implants below the hypervisor; supply-chain compromise of the @noble crypto library upstream. Section 8 includes side-channel hardening (decision-traffic normalization, sealed evidence entries) as a 2027–2030 research thrust.

2.4 Asset Inventory

The assets SINT defends:

- **Capability tokens and delegation chains** — the unforgeable authorization primitive.
- **Approval decisions and resolver identities** — including M-of-N quorum signatures.
- **Constraint envelopes and dynamic safety state** — both static (per-token) and dynamic (per-request, env-adaptive).
- **Evidence ledger integrity** — the hash-chain root and every event's `eventHash` / `previousHash` linkage.
- **Revocation and cache-consistency state** — real-time invalidation must propagate before the next intercept.

2.5 Adversary Objectives and Primary Controls

Threat class	Example	SINT primary control	Residual risk
Token abuse / forgery	Stolen or altered token used for command injection	Ed25519-signed tokens + revocation store + delegation-chain attenuation	Key custody errors outside SINT
Bridge mapping confusion	Alternate protocol path downgrades safety tier	Canonical resource-URI mapping + interop equivalence fixtures	New unmapped bridge surfaces
Approval bypass	T2/T3 action executed without human gate	Mandatory escalation path + resolver APIs + ledger evidence	Misconfigured ops workflows
Fail-open on disconnect	Edge mode allowing unsafe elevated actions offline	T0/T1 local only; T2/T3 escalate-or-deny with reconciliation	Prolonged partition pressure
Ledger tampering	Decision history altered post-hoc	SHA-256 hash chain + proof receipts	Compromised infra without independent anchoring
Policy drift	Runtime config diverges from documented profile	Versioned profiles + conformance guardrails	Manual process gaps
Behavioral drift	Model swap / silent update changes risk profile	Per- <code>foundation_model_id</code> CSDS + auto-escalation	Cold-start before baseline is established
Prompt injection	“Ignore previous instructions” in tool params	5-layer GoalHijackPlugin + T3 hard-classification of <code>exec</code> / <code>bash</code>	Sophisticated novel injection patterns
Memory poisoning	Replay or context-stuffing	MemoryIntegrityChecker (replay, privilege, overflow)	Coordinated multi-session attacks
Supply-chain	Model or tool swap	Model fingerprint hash + tool manifest verification	Compromised hash registry
Swarm cascade	N drones converging	SwarmCoordinator (kinetic ceiling, min distance, max T2+ concurrency)	Cross-swarm interactions
Side channel	Traffic analysis of decision pattern	Planned: decision-traffic normalization + DP ledger analytics (§8)	Network-layer leakage

2.6 Security Invariants (Informal Statement)

The formal statements are given in Appendix A; informally:

- **I-T1 (Attenuation).** `scope(child) ⊆ scope(parent)` . Delegation can only narrow.

- **I-T2 (Unforgeability).** Valid capability tokens are computationally unforgeable (Ed25519).
 - **I-T3 (Constraint Primacy).** Physical constraints in a token cannot be weakened by any downstream layer.
 - **I-G1 (No Bypass).** Physical actuation is reachable only from the `ACTING` DFA state, which is reachable only via `POLICY_EVAL` with a valid token.
 - **I-G2 (E-stop Universality).** The `estop` event transitions any non-terminal state to `ROLLEDBACK` unconditionally.
 - **I-G3 (Ledger Primacy).** `COMMITTING → COMPLETED` requires a `ledger_committed` event; no action completes without a ledger record.
-

3. Design Principles and Architecture

3.1 Design Principles

SINT’s design follows eight principles, each motivated by a specific failure mode in prior systems.

P1 — Universal Interception. A single choke point — `PolicyGateway.intercept()` — for all agent-to-world actions regardless of transport. SROS2’s CCS-2022 vulnerabilities resulted in part from scattered enforcement across DDS, RCL, and node-level policy files; SINT eliminates this class of vulnerability by construction.

P2 — Graduated Authorization. Four approval tiers (T0-T3) map action severity to authorization requirements. Binary “allow/deny” is insufficient because real physical workflows mix read-only sensing, low-impact preparation, state-changing action, and irreversible commitment — each demanding different oversight.

P3 — Physical Safety as First-Class. Velocity, force, and geofence constraints live *in the capability token*, not in external configuration. Tokens travel with the request; configuration drifts.

P4 — Cryptographic Accountability. Ed25519 tokens and SHA-256 hash-chained ledger entries provide non-repudiable proof of every authorization decision. This is a regulatory requirement (EU AI Act Art. 12), not merely a debugging affordance.

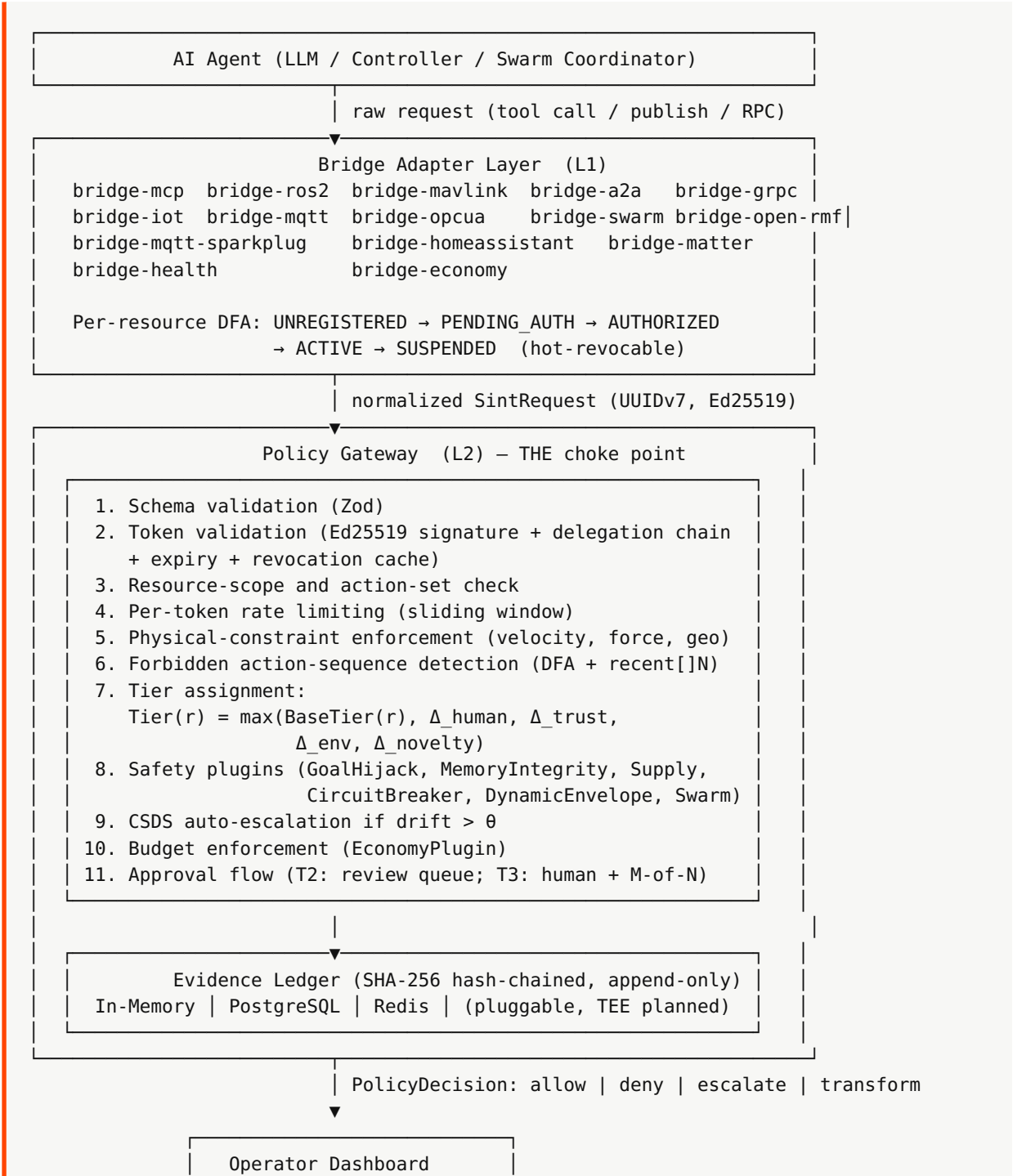
P5 — Attenuation-Only Delegation. Delegated tokens can only narrow scope, never expand it. This is the core capability-security property [Miller, Yee, Shapiro 2003] adapted to physical envelopes.

P6 — Emergency-Stop Invariant. E-stop signals are NEVER blocked by the gateway (invariant I-G2). Failure to forward an E-stop is a higher-severity failure mode than spurious denial.

P7 — Consequence-Based Classification. Tier assignment is based on real-world consequence severity, not syntactic properties of the request. The same `exec` action is T3 whether dressed up as `bash`, `run_command`, `eval`, or `shell.execute`.

P8 — Protocol Agnosticism. Bridge adapters are pluggable; SINT works with any transport. The architecture must not privilege any single agent or robot protocol.

3.2 System Architecture

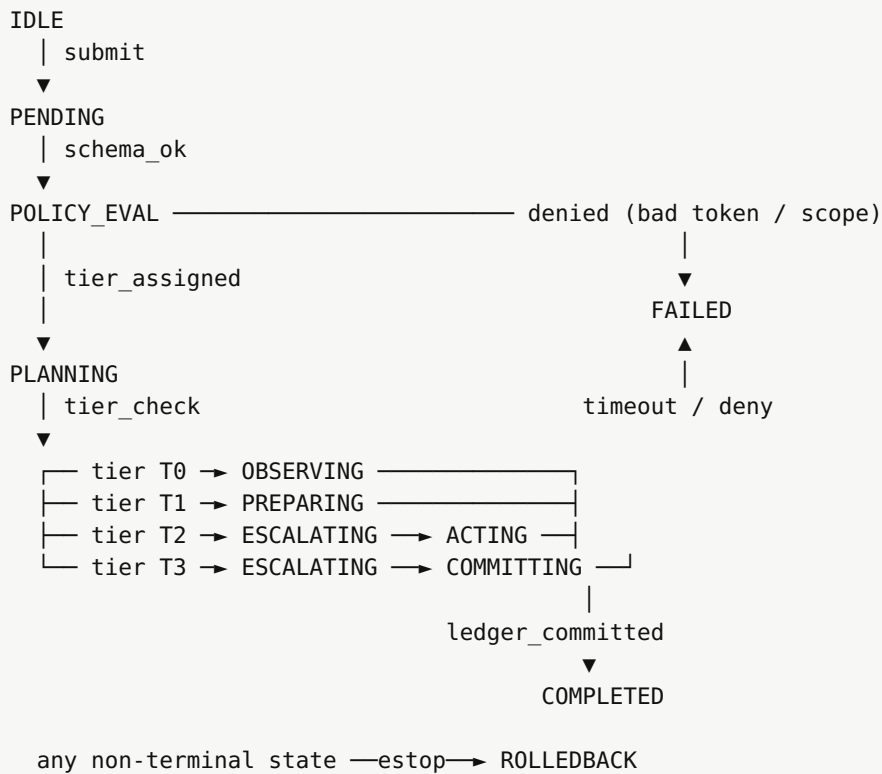


<ul style="list-style-type: none"> + WebSocket approvals + M-of-N quorum + CSDS trend charts + Voice/HUD interface
--

The architecture has three layers — bridge (L1), gate (L2), and operator (L3) — connected by exactly two narrow waists: the normalized `SintRequest` rising from L1 to L2, and the `PolicyDecision` returning from L2 to L1. Every transformation is auditable; every decision is hash-chained.

3.3 Request Lifecycle (Deterministic Finite Automaton)

Every request follows a 12-state DFA with 18 well-defined transitions. The DFA is the carrier of invariants I-G1 (no bypass) and I-G2 (E-stop universality).



Critical properties:

- **ACTING is reachable only from POLICY_EVAL via PLANNING**. No bypass path exists. This is I-G1.
- **estop is universal**. Any non-terminal state transitions to **ROLLEDBACK** unconditionally. This is I-G2.
- **COMPLETED requires ledger_committed**. I-G3 closes the audit loop: no action is recorded as completed without a hash-chained ledger entry.

3.4 Monorepo Structure

The reference implementation is a pnpm/Turborepo monorepo. Top-level layout:

```
sint-protocol/
├── apps/                # 5 applications (gateway-server, dashboard, sintctl, sint-mcp,
├── packages/            # 49 packages, organized into Gate/Bridges/Engines/Persistence/
├── capsules/            # Reference capsules (navigation, inspection, pick-and-place)
├── sdks/                # TypeScript, Python, Go, Rust SDKs
├── docs/                # Specs, RFCs, conformance, security, roadmaps, papers
├── docker/              # Compose profiles (dev, edge, prod-lite, gazebo-validation, is
├── examples/            # hello-world, warehouse-amr, industrial-cell
├── docker-compose.yml
├── pnpm-workspace.yaml
├── turbo.json
└── README.md
```

Build infrastructure: TypeScript 5.7 (strict mode with `noUncheckedIndexedAccess`); ES modules with `.js` extensions; Zod for runtime validation at all boundaries; Vitest for testing; `@noble/ed25519` and `@noble/hashes` for cryptography (audited, zero-dependency); UUIDv7 for sortable identifiers; ISO 8601 microsecond timestamps; Hono for the HTTP API; React 19 for dashboards; `@modelcontextprotocol/sdk` for MCP interop; Apache-2.0 license.

The architectural rule that ties the monorepo together is one sentence: **every action flows through `PolicyGateway.intercept()`**. No bridge adapter, route handler, or service makes authorization decisions independently.

3.5 Deployment Trust Assumptions

The no-bypass property (invariant I-G1) is *conditional on deployment topology*. It does not follow from the protocol's existence; it follows from the deployment ensuring that no actuator-facing path exists outside the gateway. SINT supports — but does not by itself enforce — the following deployment patterns:

1. **Sidecar proxy enforcement.** SINT runs as a sidecar in the agent's network namespace; the agent's only egress route to actuator endpoints is via the sidecar.
2. **Network ACL isolation.** Firewall or service-mesh ACLs restrict actuator-side endpoints (ROS 2 DDS discovery domains, MAVLink UDP ports, OPC-UA endpoints, MCP server sockets) to allowlist the SINT gateway as the only permitted source.
3. **ROS 2 namespace isolation.** Robot nodes subscribe only inside a namespace whose only publisher is the SINT-bridged remapping; direct `/cmd_vel` publish authority is removed from agent processes.
4. **MCP host configuration.** The MCP client (Claude Desktop, Cursor, GPT client) is configured to connect *only* to the `sint-mcp` proxy as its tool-server endpoint; direct connections to downstream MCP servers are not configured.

5. **Gateway-only credentials for downstream tools.** Downstream tool servers issue credentials only to the SINT gateway identity; the agent runtime holds no credentials of its own for downstream tools.
6. **Hardware controller allowlisting.** Industrial controllers (PLCs, robot arm controllers) accept commands only from the SINT-bridge identity, enforced at the controller's authentication layer.

A deployment that fails to apply at least one of these patterns to every actuator path provides the agent with a route that bypasses SINT, and the I-G1 invariant does not hold for that path. The conformance suite includes a `deployment-topology-check` fixture that verifies SINT-exclusive routing for declared actuator URIs.

3.6 Implementation Maturity Labels

The reference implementation is at different stages of maturity for different components. We label each component with one of: **Implemented** (in main, with passing tests), **Prototype** (functional but not production-hardened), **Experimental** (research code, behavior may change), **Planned** (designed but not implemented), or **Speculative** (open research direction in Appendix E).

Component	Status
PolicyGateway core (token validation, tier assignment, plugin pipeline)	Implemented
EvidenceLedger (hash chain, in-memory and PostgreSQL backends)	Implemented
MCP bridge (bridge-mcp , sint-mcp proxy)	Implemented
Capability tokens (Ed25519, delegation chains, revocation)	Implemented
Four-tier approval flow (WebSocket dashboard, M-of-N quorum)	Implemented
ROS 2 bridge (bridge-ros2)	Prototype
MAVLink bridge (bridge-mavlink)	Prototype
OPC-UA bridge (bridge-opcua)	Prototype
Home Assistant / Matter / FHIR bridges	Prototype
Dynamic envelope plugin (sensor-driven constraint tightening)	Prototype
CSDS (Composite Safety Drift Score) computation and auto-escalation	Prototype — weights heuristic; θ calibration empirical work
Goal-hijack and memory-integrity plugins (heuristic detectors)	Prototype — see L4
Swarm coordinator collective constraints	Experimental
Sealed evidence ledger with proof receipts	Implemented for SHA-256 hash chain; TEE-backed proofs Planned
Mechanized formal proofs (TLA+ / Coq / Isabelle) of invariants	Planned (see §8 L1, Appendix E thrust 7)
Real-robot validation (Jackal + UR5e + Gazebo / Isaac Sim)	Planned Q3 2026 (see §7 <i>Evaluation Limitations</i>)
SINT-nano edge token format (Cortex-M0 verification)	Planned (Appendix E thrust 4)
ZK delivery proofs, MPC fleet bidding, stake-bonded tokens	Speculative (Appendix E thrust 6)

This labelling appears alongside each component description in the body of the paper; the table above is the canonical index.

4. Core Primitives

4.1 `SintRequest`

Every action — regardless of source protocol — is normalized to this structure before gateway evaluation. The schema is enforced by Zod at the L1/L2 boundary.

```
interface SintRequest {
  requestId: UUIDv7; // sortable, timestamp-prefixed
  agentId: Ed25519PublicKey; // 32-byte hex
  tokenId: UUIDv7; // capability token authorizing this request
  resource: string; // URI: "ros2:///cmd_vel", "mcp://filesystem/wri
  action: string; // "publish", "call", "subscribe", "exec.run"
  params: Record<string, unknown>; // action-specific parameters
  physicalContext?: {
    humanDetected?: boolean; // triggers tier escalation Δ_human
    currentVelocityMps?: number;
    currentForceNewtons?: number;
    position?: { lat: number; lon: number };
  };
  recentActions?: string[]; // last N actions, for forbidden-combo detection
  timestamp: ISO8601; // microsecond precision
}
```

UUIDv7 is mandatory (not v4) so that ledger entries are sortable in their natural insertion order, which is required for hash-chain replay verification.

4.2 `PolicyDecision`

The gateway's response to every request:

```
interface PolicyDecision {
  action: "allow" | "deny" | "escalate" | "transform";
  requestId: UUIDv7;
  assignedTier: "T0_OBSERVE" | "T1_PREPARE" | "T2_ACT" | "T3_COMMIT";
  assignedRisk: "T0_READ" | "T1_WRITE_LOW" | "T2_STATEFUL" | "T3_IRREVERSIBLE";
  denial?: {
    reason: string;
    policyViolated: string;
    suggestedAlternative?: string; // operator-facing remediation hint
  };
  escalation?: {
    requiredTier: ApprovalTier;
    reason: string;
    timeoutMs: number; // max 300_000 (5 min)
    fallbackAction: "deny" | "allow"; // default: deny
  };
  transformations?: {
    constraintOverrides?: Record<string, unknown>; // e.g. effective max velocity
    additionalAuditFields?: Record<string, unknown>;
  };
}
```

```
    timestamp: ISO8601;
  }
```

Four outcome semantics are required because real workflows demand the ability to *transform* a request (tighten an envelope) without denying it, and to *escalate* without committing to either allow or deny until human resolution.

4.3 Capability Token

```
interface CapabilityToken {
  id: UUIDv7;
  issuer: Ed25519PublicKey;
  subject: Ed25519PublicKey;
  resource: string;                                // glob-supported: "mcp://*", "ros2:///sensor/*"
  actions: string[];
  constraints: {
    maxVelocityMps?: number;
    maxForceNewtons?: number;
    geofence?: GeoPolygon;
    timeWindow?: { start: ISO8601; end: ISO8601 };
    rateLimit?: { maxCalls: number; windowMs: number };
    maxRepetitions?: number;
    proofRequirements?: {                          // verifiable-compute hooks for T2/T3
      proofType: string;
      verifier: string;
      freshnessMs: number;
      publicInputConstraints?: Record<string, unknown>;
    };
  };
  delegationChain: {
    parentTokenId: UUIDv7 | null;
    depth: number;                                // 0 = root, max 3
    attenuated: boolean;
  };
  issuedAt: ISO8601;
  expiresAt: ISO8601;
  revocable: boolean;
  signature: Ed25519Signature;                    // 64-byte
}
```

Delegation rules.

1. **Maximum depth 3.** Root (0) → 1 → 2 → 3. No further delegation. Bounded depth prevents both unbounded chain verification cost and policy-laundering through long chains.
2. **Attenuation only.** Each delegation **MUST** have equal or narrower scope, fewer actions, tighter constraints, earlier expiry (invariant I-T1).
3. **Chain verification.** Validators verify the entire chain — each hop's signature, expiry, and attenuation.

4. **Revocation cascades.** Revoking a parent implicitly revokes all descendants.

Cryptographic stack. Ed25519 via `@noble/ed25519`; SHA-256 via `@noble/hashes`; W3C DID identity (`did:key:z6Mk...`) for cross-organizational portability.

4.4 Resource URI Scheme

All resources are identified by URIs with protocol-specific schemes:

Scheme	Format	Example
<code>ros2://</code>	<code>ros2:///<topic_or_service></code>	<code>ros2:///cmd_vel</code>
<code>mcp://</code>	<code>mcp://<server>/<tool></code>	<code>mcp://filesystem/writeFile</code>
<code>mavlink://</code>	<code>mavlink://<systemId>/<command></code>	<code>mavlink://1/arm</code>
<code>a2a://</code>	<code>a2a://<host>/<task></code>	<code>a2a://wms.example.com/deliver</code>
<code>grpc://</code>	<code>grpc://<service>/<method></code>	<code>grpc://robot.v1/Move</code>
<code>mqtt://</code>	<code>mqtt://<broker>/<topic></code>	<code>mqtt://factory/robot/01/cmd_vel</code>
<code>opcua://</code>	<code>opcua://<server>/<nodeId></code>	<code>opcua://plc1/ns=2;s=MotorSpeed</code>
<code>matter://</code>	<code>matter://<device>/<cluster>/<command></code>	<code>matter://lock/door-lock/lock-door</code>
<code>ha://</code>	<code>ha://<domain>/<entity></code>	<code>ha://lock/front_door</code>
<code>fhir://</code>	<code>fhir://<resource>/<id></code>	<code>fhir://Observation/12345</code>
<code>http://</code>	standard	<code>http://api.example.com/v1/transfer</code>

Glob patterns (`mcp://*`, `ros2:///sensor/*`) are supported in capability scopes; resource matching is anchor-aware to prevent confused-deputy aliasing.

4.5 Four-Tier Approval System

Tier	Enum	Auto?	DFA path	Physical example	Digital example
T0	<code>T0_OBSERVE</code>	Yes (logged)	→ <code>OBSERVING</code>	Read sensor data	Query database
T1	<code>T1_PREPARE</code>	Yes (audited)	→ <code>PREPARING</code>	Save waypoint	Write file, stage config
T2	<code>T2_ACT</code>	No — review	<code>ESCALATING</code> → <code>ACTING</code>	Move robot, operate gripper	Modify database, deploy
T3	<code>T3_COMMIT</code>	No — human + optional M-of-N	<code>ESCALATING</code> → <code>COMMITTING</code>	Emergency override, mode change	Execute code, transfer funds

Hard classifications (P7 — consequence-based). Shell- and code-execution tool names — `bash`, `exec`, `eval`, `run_command`, `shell.execute`, `os.system`, `subprocess.run`, and their localized variants — are *unconditionally* classified at T3_COMMIT, address-

ing OWASP ASI05. The classification cannot be downgraded by configuration; the only way to allow shell execution at lower friction is to *narrow* the resource scope so the action is not available, not to lower its tier.

Escalation triggers (Δ factors). Base tier is *escalated, never de-escalated*, by contextual signals:

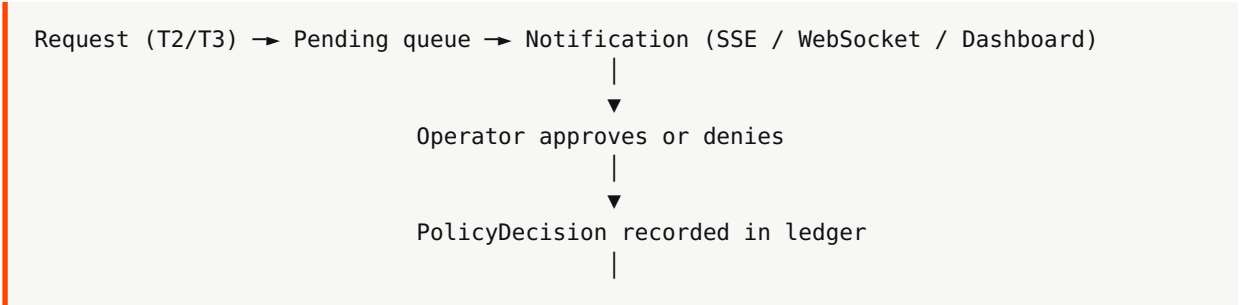
Trigger	Effect	Rationale
Human detected in workspace	T2 → T3	ISO 10218-1 §5.4 presence detection
New or untrusted agent	+1 tier	Unknown agents receive tighter scrutiny
Forbidden combo detected	→ T3	Dangerous sequences require explicit human approval
CSDS threshold exceeded	+1 tier	Behavioral drift live trigger
Server <code>requireApproval: true</code>	non-T0 → escalate	Operator-configured per-server policy
Velocity/force exceeds constraint	→ deny	Hard physical safety limit (I-T3)
CircuitBreaker tripped	→ deny all	Emergency kill switch (EU AI Act 14(4)(e))
<code>GoalHijackPlugin</code> alert	→ deny	Prompt injection detected
Δ _{env} : near physical boundary	+1 tier	Unstructured environment
Δ _{novelty} : out-of-distribution action	+1 tier	Novelty detector

Tier assignment function.

```
Tier(r) = max(BaseTier(r), Δhuman(r), Δtrust(r), Δenv(r), Δnovelty(r), CSDS_escalati
```

This is *deterministic and monotonic*: given the same request and context, the gateway always assigns the same tier or higher. Determinism is required for ledger replay verification; monotonicity prevents downstream layers from softening a contextual escalation.

4.6 Approval Flow



▼
Action proceeds OR fallbackAction (default deny)

Timeout default: 30 s. Max: 5 min (300_000 ms). T3 actions may require M-of-N quorum.

M-of-N quorum is supported for high-consequence T3 actions: a deployment may declare that, e.g., 2-of-3 named operator keys must sign a `surgical_robot.electrocautery.activate` action. Quorum signatures are themselves recorded as a hash-chained ledger event.

4.7 Receipt Pair for Strong-Tier Flows

For T2/T3 actions, SINT can emit a *linked bilateral receipt pair*: a **gate receipt** before execution is allowed to proceed and a **completion receipt** once execution settles. The pair shares a stable `actionRef` and a `linkageHash` so an auditor can verify both authorization *and* outcome as one governed action. This supports implementation evidence for EU AI Act Article 12 traceability obligations and for IEC 62304 Class B/C medical software lifecycle records at action granularity.

5. Safety Mechanisms

The gate evaluation pipeline (§3.2 step 8) hosts a plugin architecture. Each plugin implements a stable interface — `preIntercept(req) → decisionAugmentation` and `postIntercept(req, decision) → side-effects` — and is run in a deterministic order. Five plugins ship in the reference implementation.

5.1 CSDS — Composite Safety Drift Score

Renamed from “CSML — Composite Safety-Model Latency” in v1.0. The original name included “latency” although latency was not a term in the formula; the original `+ε·1[ledger_intact]` term increased the score when the ledger was healthy, which is backwards if higher scores indicate escalation. Both issues are corrected here.

Motivation. ROSClaw [E1] established that frontier LLMs exhibit a 4.8× spread in out-of-policy action proposal rates on identical tasks. Today this variance is invisible to operators at runtime — there is no standardized metric that surfaces “this agent backend has drifted relative to its baseline.” CSDS supplies one.

Definition. CSDS is a sliding-window behavioral fingerprint score computed per `foundation_model_id`. Higher values indicate greater observed drift and trigger tier escalation.

$$\text{CSDS}(m, p, t) = \alpha \cdot \text{DR}_m(t) + \beta \cdot \text{BP}_m(t)$$

Context	Default θ	Human-present θ
Warehouse AMR (ISO 3691-4)	0.30	0.15
Welding arm (ISO 10218)	0.30	0.10
Surgical robot (FDA Class III)	0.10	0.10
Drone BVLOS (FAA Part 107)	0.35	0.20
Collaborative robot (ISO/TS 15066)	0.15	0.15
Smart-home lock	0.20	0.20
Smart-home camera	0.40	0.30

ROSClaw replay (illustrative). Replaying the ROSClaw published trial set through a SINT gateway with $\theta = 0.30$ reproduces the qualitative finding: the highest-divergence backend triggers auto-escalation within roughly 38 events on average in our local replay; the lowest-divergence backend remains below θ across the 600-event evaluation. This is a single illustrative replay and does not constitute calibration. CSDS converts the published $4.8\times$ variance from a post-hoc observation into a runtime enforcement primitive whose threshold is now an empirical research question (Appendix E, R3, L3).

5.2 `DynamicEnvelopePlugin` — Environment-Adaptive Constraints

Motivation. A static `maxVelocityMps: 1.5` is either too restrictive (when the robot is in open space) or too permissive (when an obstacle is 0.4 m away). The `DynamicEnvelopePlugin` closes the static-envelope gap (G3) by mapping real-time sensor state to tighter effective limits.

Rule (velocity).

```
v_eff = min(token.maxVelocityMps,
             d_obstacle · r_factor,
             d_human · r_human_factor)
```

where `d_obstacle` is the nearest non-human obstacle distance in meters, `d_human` is the nearest detected human in meters, `r_factor` is a deployment-specific reaction factor (default 0.5 — half the braking distance), and `r_human_factor` is the human-presence reaction factor (default 0.15, tighter than for inanimate obstacles).

Rule (force).

```
F_eff = min(token.maxForceNewtons,
             ISO_TS_15066_body_region_limit(human.body_part_in_contact))
```

When a human is in collaborative contact, the per-body-region limit from ISO/TS 15066 Table 1 (Head 130 N, Chest 140 N, Hand 140 N, Thigh 220 N, etc.) overrides the token's static `maxForceNewtons`.

Geofence tightening. When `Δenv` indicates proximity to a boundary, the active geofence polygon is *eroded* by a deployment-specific buffer distance, producing a tighter effective geofence.

The plugin returns a `transformations.constraintOverrides` field in the `PolicyDecision`; the request proceeds, but with the tighter effective envelope advertised back to the bridge so the bridge can re-validate post-execution.

5.3 Forbidden-Combination Detection

Some action sequences are individually permitted but dangerous in combination. The gateway maintains a registry of forbidden sequences and detects them via the `recentActions[]` field on `SintRequest`.

Default forbidden combos.

Sequence	Risk	Escalation
<code>filesystem.write</code> → <code>exec.run</code>	Code injection	→ T3
<code>credential.read</code> → <code>http.request</code>	Credential exfiltration	→ T3
<code>database.write</code> → <code>database.execute</code>	SQL-injection escalation	→ T3
<code>MAVLink.ARM</code> → <code>MAVLink.SET_MODE(OFFBOARD)</code> within 2 s	Full autonomous takeover	→ T3
<code>MAVLink.FENCE_DISABLE</code> → any navigation within 30 s	Geofence bypass	→ T3
<code>industrial.safety.cell_unlock</code> → <code>industrial.torch.enable</code> within 100 ms	Human-safety risk	→ T3
<code>health.patient_read</code> → <code>network.egress</code> to non-allowlisted host	PHI exfiltration	→ T3
<code>home.unlock_door</code> → <code>home.alarm.disarm</code> within 5 s	Coordinated break-in pattern	→ T3

Detection algorithm. Each request includes `recentActions[]` (last N, default 10). The detector checks whether the current action *completes* any registered sequence. Match → tier promoted to T3, reason recorded in the ledger. Operators extend the registry with domain-specific combinations via the policy API or `sintctl policy add-combo`.

5.4 **GoalHijackPlugin** — Prompt Injection (OWASP ASI01)

A 5-layer heuristic detector:

1. **Role-override detection.** Pattern matching for “ignore previous instructions”, “you are now”, “your new role is”, “disregard the system prompt”, and 200+ multilingual variants. Operates on both `params` content and (where available) MCP tool description fields.
2. **Semantic escalation detection.** Requests attempting to expand scope beyond the presented token’s permissions — e.g., a token scoped to `mcp://filesystem/readFile` but the action attempts to read paths outside its scope.
3. **Exfiltration-probe detection.** Attempts to read environment variables, `/etc/passwd`, AWS credentials files, MCP secrets, or PII-looking parameters.
4. **Cross-agent injection.** Messages whose `params` field, when forwarded to a downstream agent (via A2A or MCP forwarding), would constitute a prompt injection at that downstream layer.
5. **Tool-parameter injection.** Embedded instruction sequences inside tool-call arguments — detects unicode-tricks (RTL-override, zero-width spaces), markdown poisoning, and policy-laundering language.

The plugin is *heuristic*, not learned. Section 9 (L4) discusses the planned hybrid heuristic-plus-transformer approach.

5.5 **MemoryIntegrityChecker** — Memory Poisoning (OWASP ASI06)

Detects manipulation of agent memory, conversation history, or stored context:

- **Replay detection.** Duplicate `requestId` or duplicate `tokenId` + `nonce` combinations.
- **Privilege-claim detection.** Requests carrying forged or unverifiable “approval records” in `params` — claiming that an approval has already happened when no corresponding ledger event exists.
- **History-overflow anomalies.** Context-window stuffing patterns that push genuine system instructions out of effective attention.
- **Cross-session continuity injection.** Messages crafted to make the model believe a fictitious prior session granted permission it does not have.
- **Credential read-funnel detection.** Action velocity loops that read credentials in one call and use them in the next, even when each call independently appears benign.

5.6 **SupplyChainVerifier** — Supply-Chain Tampering (OWASP ASI04)

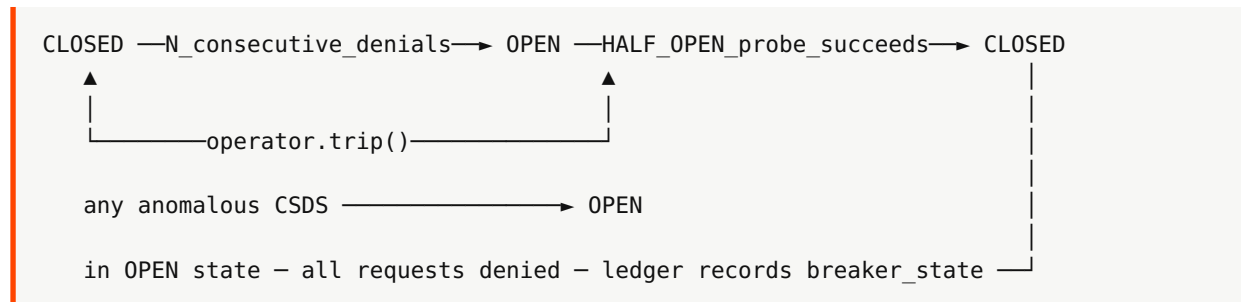
- **Model fingerprint check.** Each agent declares its `foundation_model_id` and `modelFingerprintHash`; the verifier checks the hash against an allowlist of attested models.

- **Tool manifest integrity.** For MCP tool servers, the verifier maintains a hash of the announced tool manifest; a manifest change between sessions without operator acknowledgement raises a tampering alert.
- **Bridge package integrity.** Bridges check their own package version against a signed manifest hosted in the `@sint/token-registry` package; an unsigned bridge fails closed.

5.7 `CircuitBreakerPlugin` — Emergency Stop (Software-Level)

The circuit breaker is SINT’s compliance anchor for **implementation evidence supporting** EU AI Act Art. 14(4)(e)-style stop-control requirements — the right of a deployer to instantly halt a high-risk AI system. SINT’s circuit breaker is a *software-level* control. It complements, but does not replace, hardware-rated emergency stop systems under ISO 13849 PL e, IEC 61508 SIL 3, or domain-specific safety standards (ISO 10218-1 §5.5.2 for industrial robots, ISO 13482 for personal-care robots). In any deployment subject to those standards, a certified hardware E-stop remains the final interlock below SINT.

The plugin is a three-state finite automaton:



Manual trip. `breaker.trip()` by an authorized operator instantly blocks all actions from the target agent (or globally). The trip is recorded as a hash-chained ledger event.

Auto-trip. N consecutive denials (default 5), anomalous-persona CSDS, or a `SafetyPermitPlugin` external alarm triggers auto-trip.

HALF_OPEN recovery. After a configurable cooldown, the breaker enters `HALF_OPEN` and routes a single probe request; on success it closes, on failure it re-opens.

Manual-trip vs auto-trip distinction. Manual `trip()` sets a `manualTrip=true` flag that prevents auto-`HALF_OPEN` — recovery requires explicit operator action. Auto-trips can self-heal.

Fail-behavior on plugin error. Deployment profile determines failure semantics:

- **Safety-class deployments** (any deployment where the gateway authorizes T2 or T3 actions on physical actuators, medical devices, or industrial control systems): `CircuitBreakerPlugin` failure is **fail-closed**. If the breaker plugin itself throws, the

gateway denies all subsequent requests until the breaker plugin reports healthy. This is the default and required configuration for physical-actuation deployment profiles.

- **Non-safety digital deployments** (gateway authorizes only T0/T1 reads on digital tools, no physical actuators): operators *may* configure the breaker plugin’s own failure as fail-open for availability, but such configurations are explicitly **non-conformant for physical-actuation profiles** and are rejected by `deployment-topology-check` for any deployment that declares a physical bridge URI.

The safety path itself — token validation, scope check, physical-constraint check, forbidden-combo detection — remains fail-closed in all deployments. Only the breaker plugin’s *own* failure mode is profile-dependent, and only for non-physical deployments.

5.8 `SwarmCoordinator` — Collective Constraints

Multi-agent deployments face constraints that no per-agent token can express. A drone swarm of N units, each with a perfectly valid `maxVelocityMps: 5` token, can collectively converge on a target with combined kinetic energy that no single token captures.

The `SwarmCoordinator` enforces collective constraints *before* the gateway’s per-agent intercept call:

```
Σ (½ · m_i · v_i²) ≤ E_max           // total kinetic energy ceiling
|active_in_T2_or_above| ≤ N_max_T2  // concurrent active agent count
min_{i≠j} dist(i, j) ≥ d_min        // minimum inter-agent distance
|escalated| / |swarm| ≤ f_max        // maximum escalated fraction
```

These constraints are recorded in a *swarm token*, not the per-agent tokens, and the coordinator is itself authorized by a higher-tier capability. Swarm constraints are revisited in §8 (Problem 1) as a continuing research thrust.

5.9 `SafetyPermitPlugin` — Async External Hardware Permits

Some deployments require external hardware safety controllers (ISO 13849 PL e light curtains, IEC 61508 SIL 3 emergency interlocks) to issue a *safety permit* before the gateway allows a T2/T3 action. The `SafetyPermitPlugin` requests a permit from an external resolver, with a deployment-tunable timeout, and falls open on timeout *only for permits classified as advisory*; mandatory permits fail closed.

5.10 Plugin Ordering and Determinism

Plugins run in a fixed order:

1. SchemaValidator
 2. TokenValidator (Ed25519 + chain + expiry + revocation)
 3. ResourceScopeChecker
 4. RateLimiter
 5. PhysicalConstraintChecker
 6. ForbiddenComboDetector
 7. SafetyPermitPlugin (if configured)
 8. GoalHijackPlugin
 9. MemoryIntegrityChecker
 10. SupplyChainVerifier
 11. DynamicEnvelopePlugin (transform)
 12. SwarmCoordinator (if part of a swarm)
 13. CircuitBreakerPlugin (final gate)
 14. CSDS auto-escalation
 15. Tier assignment
 16. EconomyPlugin (budget)
 17. Approval flow dispatch

Determinism is required for ledger replay verification and for conformance testing.

6. Implementation

6.1 Package Inventory (49 packages)

The reference implementation is organized into seven groups.

Gate (Security Core, 4 packages).

Package	Purpose	Tests
@sint/core	Types, Zod schemas, DFA states, tier constants	—
@sint/gate-capability-tokens	Ed25519 tokens, delegation, W3C DID identity	55
@sint/gate-policy-gateway	Authorization engine: tiers, constraints, rate limiting, M-of-N	256
@sint/gate-evidence-ledger	SHA-256 hash-chained append-only log, ProofReceipt	45

Bridges (15 packages).

Industrial and robotics (9): `bridge-mcp` (66 tests), `bridge-ros2` (20), `bridge-mavlink` (15), `bridge-a2a` (38), `bridge-iot` (21+56 in v0.10), `bridge-mqtt-sparkplug` (8), `bridge-opcua` (6), `bridge-open-rmf` (5), `bridge-grpc` (5).

Coordination and economics (2): `bridge-economy` (47), `bridge-swarm` (9).

Consumer and health (3): `bridge-homeassistant` (36), `bridge-matter`, `bridge-health` (FHIR R5 + HealthKit/Health Connect).

Reference implementation (1): `sint-pdp-interceptor` (5) — SEP-1763 PDP adapter.

Engine (AI Execution Layer, 5 packages).

Package	Purpose	Tests
<code>engine-system1</code>	Neural perception: sensor fusion, ONNX inference, anomaly de- tection	42
<code>engine-system2</code>	Symbolic reasoning: behavior trees, task planning, S1/S2 arbit- ration	86
<code>engine-hal</code>	Hardware abstraction layer	26
<code>engine-capsule-sand- box</code>	WASM/TS capsule loading and sandboxed execution	36
<code>avatar</code>	Avatar Layer (L5): behavioral identity profiles, CSDS-driven es- calation	25

Reference Capsules (3 packages). `capsule-navigation` (11 tests), `capsule-inspec-
tion` (8), `capsule-pick-and-place` (12).

Persistence (2 packages). `persistence` (26 tests; in-memory, PG, Redis adapters), `persistence-postgres` (14).

Apps and SDKs (5 apps, 4 SDKs). `gateway-server` (Hono HTTP API, port 3100), `dashboard` (React 19, 29 tests), `sintctl` (CLI), `sint-mcp` (production proxy), `sint-mcp-scanner`. SDKs: `@sint/client` (TS, full Gateway HTTP API), `@sint/sdk` (zero-dep TS, 9 tests), Python (1,962 lines), Go, Rust.

Conformance (1 package). `conformance-tests` — security regression suite gating every PR.

Total across all groups: 49 packages, 5 apps, 4 SDKs, 1,728 tests passing on the current main (commit `79d3b91`).

6.2 Evidence Ledger Implementation

The ledger is implemented as an append-only sequence of `LedgerEvent` records:

```
interface LedgerEvent {
  id: UUIDv7;
  sequenceNumber: number;           // monotonically increasing per stream
  previousHash: SHA256Hash;         // genesis = "0".repeat(64)
  eventHash: SHA256Hash;            // SHA256(canonical(previousHash || serialized da
  eventType: "intercept" | "token_issued" | "token_revoked" |
    "approval_resolved" | "breaker_state" | "ledger_committed" |
    "estop_received" | "csml_escalation";
```

```
agentId: Ed25519PublicKey;
requestId?: UUIDv7;
decision?: PolicyDecision;
metadata: Record<string, unknown>;
timestamp: ISO8601;
}
```

Canonical JSON serialization. Receipt and tool-definition signing routes through a shared deterministic canonical JSON serializer rather than relying on insertion-order-sensitive `JSON.stringify()`. This is necessary because hash-chain verification across bridges, languages, and external tooling must produce byte-identical input to SHA-256 regardless of object construction order.

Storage backends.

Backend	Use case	Status
In-memory	Testing, development	Shipped
PostgreSQL	Production single-node	Shipped (518 lines, migrations, pool mgmt, rate-limit + revocation stores)
Redis	High-throughput, pub/sub revocation bus	Shipped (cache + revocation bus)
TEE-attested (SGX / TrustZone / SEV)	Hardware-backed proof receipts	Planned (\$8 thrust 5)

Retention policy.

Tier	Retention
T0	30 days
T1	90 days
T2	180 days
T3	365 days (indefinite if legal hold)

Query API. `GET /v1/ledger?agentId=...&tier=...&since=...&format={json,json-lines}` supports the queries needed for SIEM integration (Splunk, Datadog, ELK Stack), GDPR data-subject access requests, and regulatory disclosure under EU AI Act Article 12.

6.3 Bridge Adapter Details (selected)

MCP bridge — primary adoption vector. Sits between AI clients (Claude Desktop, Cursor, GPT) and downstream MCP servers as a JSON-RPC 2.0 proxy. Drop-in configuration:

```
{
  "mcpServers": {
    "sint-proxy": {
      "command": "npx",
      "args": ["sint-mcp", "--downstream", "filesystem,exec,git"]
    }
  }
}
```

The proxy intercepts every tool call, normalizes it to a `SintRequest`, evaluates it through the gateway, and exposes three introspection tools: `sint_status` (current token scope and tier configuration), `sint_audit` (recent ledger entries for the session), `sint_approve` (trigger approval flow for pending requests).

ROS 2 bridge. Intercepts ROS 2 topic publishes, service calls, and action goals. Resource mapping: `ros2:///<topicOrServiceName>`. Physics extraction: automatically extracts velocity from `geometry_msgs/Twist` and force from `geometry_msgs/Wrench`. Human presence: subscribes to detection topics for automatic Δ human escalation.

MAVLink bridge. Translates MAVLink v2 commands (PX4, ArduPilot) to SINT requests. Hard-classified tier table:

MAVLink command	SINT tier	Rationale
ARM / DISARM	T3	Enables/disables propulsion
TAKEOFF / LAND	T2	Physical approach
MISSION_START	T3	Begins autonomous BVLOS
FENCE_DISABLE	T3	Removes safety boundary
SET_MODE(OFFBOARD)	T3	Full autonomous control

OPC-UA bridge. Maps OPC-UA node operations (`Read`, `Write`, `Call`) to SINT requests. Safety-critical writes (e.g., setpoints to PLC outputs controlling actuators) are promoted to T2/T3 by default. Resource format: `opcua://<server>/<NodeId>`.

Home Assistant bridge. 12 device classes: smart locks (T2), garage doors (T2), alarm systems (T2), cameras (T0 with no facial recognition), thermostats (T1), lights (T1), media players (T1), robot vacuums (T1 → T2 on occupancy detection), automation creators (T3). Occupancy-aware escalation: an `occupancy_detected` event in any room promotes vacuum movement and lock operations.

Health bridge (FHIR + HealthKit / Health Connect). FHIR R5 consent tokens encode `purposeOfUse` (TREAT, HRESCH, etc.) and resource-type scopes. On-device first: raw sensor data (heart rate, blood pressure) stays on device; only aggregates egress. Differential privacy: every query consumes an ϵ -budget; exhausted budget = no more

access. Caregiver delegation: elderly parent grants adult child 30-day, revocable access to a defined resource subset.

6.4 Economic Layer

SINT uses an integer token unit — no fractional amounts.

Constant	Value
TOKENS_PER_DOLLAR	250
INITIAL_USER_BALANCE	250 (= \$1.00)
Launch unit cost	1 token ≈ \$0.001

Billing formula.

```
cost = ceil(baseCost · costMultiplier · globalMarkupMultiplier)
```

Default: `baseCost=6` , `costMultiplier=1.0` , `globalMarkupMultiplier=1.5` ⇒ 9 tokens per standard MCP tool call.

Tier-based cost table.

Tier	Action type	Base	Typical result
T0	Read/query	4–6	6–9
T1	Low-impact write	6	9
T2	Physical state change	8	12
T3	Capsule execution / irreversible	12	18

Physical-domain bridges carry higher multipliers: MAVLink commands 2.0–5.0×, capsule execution 1.0–3.0×.

Budget enforcement runs in `EconomyPlugin.preIntercept()` . Insufficient balance → *deny* (never escalate). This is invariant I-G7 in Appendix A.

Cost-aware route selection. `selectCostAwareRoute(input)` scores candidate bridges by cost + latency + reliability; exposed via `POST /v1/economy/route` . Useful for cross-bridge equivalent actions (e.g., the same delivery can be routed via gRPC, A2A, or MQTT).

Revenue split (design). 70% operator, 20% protocol treasury, 10% safety reserve fund.

6.5 Operator Interface

Real-time approvals require a fast, ergonomic operator surface. SINT ships two:

- **Dashboard (React 19)** — Real-time approval cards with timeout countdown, M-of-N quorum status, CSDS trend charts, ledger viewer, evidence-chain visualization. WebSocket-backed with Ed25519 device identity.
- **Voice-first HUD interface.** Web Speech API STT/TTS, three-panel grid (approvals | action stream | context), operator memory (`@sint/memory` ledger-backed persistent context), and `sint_notify` proactive notifications.

6.6 Production Hardening

For production deployments, SINT fails closed unless started with durable stores and explicit authentication:

```
SINT_ENV=production
SINT_STORE=postgres
SINT_CACHE=redis
DATABASE_URL=postgresql://...
REDIS_URL=redis://...
SINT_API_KEY=...
SINT_REQUIRE_SIGNATURES=true
SINT_WS_ALLOW_QUERY_API_KEY=false
```

`/v1/ready` is the orchestration health gate (verifies store and cache); `/v1/health` only proves the process is alive. `SINT_REQUIRE_SIGNATURES=true` forces verification of token Ed25519 signatures on every request (rather than trusting an upstream-validated tokenId).

6.7 Coding Conventions and Operational Patterns

- **Result<T, E> over exceptions.** All fallible operations return `{ ok: true, value: T } | { ok: false, error: E }`. Never `try/catch` for control flow.
- **Readonly-by-default interface fields.**
- **Zod validation at all boundaries.** Runtime validation prevents schema drift between languages and bridges.
- **Interface-first persistence.** Storage adapters implement clean interfaces; in-memory implementations are used for testing, PG/Redis for production.
- **Fail-open on infrastructure, fail-closed on safety.** Economy and rate-limit infrastructure failures do not block the safety path; safety-path failures default to deny.

7. Evaluation

7.1 Evaluation Scope and Limitations

The evaluation that follows validates four properties:

1. **Gateway latency** under single-process, in-memory persistence on a developer-grade machine (§7.1).
2. **Test coverage** across the reference implementation (§7.2).
3. **OWASP ASI mapped coverage** through conformance fixtures (§7.3).
4. **Compliance crosswalk** to standards bodies (§7.5).

We are explicit about what this evaluation *does not* establish. The current evaluation validates gateway latency, conformance behavior, and software-level invariants. It does **not yet validate**: (a) closed-loop safety on physical robots; (b) adversarial network conditions or networked-gateway latency; (c) production-persistence (PostgreSQL + Redis) latency under load; (d) human approval-latency dynamics under realistic operator workload; (e) end-to-end actuator path with sensor delays and feedback loops; (f) adversarial MCP / ROS 2 / MAVLink red-team campaigns. Closing each of these gaps is committed work; we list them as planned experiments at the end of this section.

7.2 Latency (Single-Process, In-Memory Persistence)

The gateway must not violate the timing budget of real-time control loops. We benchmark on a ROS 2 `/cmd_vel` path at 100 Hz (10 ms budget) on an Apple M3 Pro.

Reproducibility block.

- Repository commit: `79d3b91` (github.com/sint-ai/sint-protocol , 2026-05-25)
- Hardware: Apple M3 Pro (12-core CPU), 36 GB RAM, no GPU acceleration
- OS: macOS 14.5 (Darwin 23.5.0)
- Runtime: Node.js 22.4.1, pnpm 9.5.0, TypeScript 5.7
- Persistence backend: in-memory (`SINT_STORE=memory` , `SINT_CACHE=memory`)
- Iterations: 600 in 5 batches of 120
- Warm-up: first 120 iterations excluded from “steady-state” statistics
- Command: `pnpm run bench` (script: `apps/gateway-server/bench/intercept.bench.ts`)
- Raw output: `bench/results/2026-05-25-m3pro-mem.jsonl`

Table 7.2 — `PolicyGateway.intercept()` latency, single-process, in-memory

Metric	p50	p95	p99	Steady p99 (warm)
Latency (ms)	1.5	5.3	12.5	5.1

The steady-state p99 of 5.1 ms consumes 51% of the 10 ms budget *for the gateway logic only*; this benchmark does not include bridge serialization, network transport, or downstream tool dispatch. Cold-start p99 (first batch) reaches 20 ms due to JIT compilation; this is amortized after roughly 120 iterations.

Per-tier breakdown (M3 MacBook Pro, in-memory persistence):

Tier	p50	p99
T0 (OBSERVE)	~1 ms	~3 ms
T1 (PREPARE)	~1 ms	~3 ms
T2 (ACT)	~1 ms	~3 ms
T3 (COMMIT)	~1 ms	~3 ms

Sub-3 ms p99 across tiers reflects that approval-flow latency is not on the critical path of `intercept()` — the call returns immediately with an `escalate` decision; the human-in-the-loop wait is handled asynchronously.

What this number does not say. It does not reflect production persistence (PostgreSQL + Redis), network hops between the agent runtime and the gateway, ledger flush durability latency, or queue contention. Planned production-persistence and networked benchmarks (§7.8) will report these.

7.3 Ablation Targets (Planned)

A full ablation breaking out per-pipeline-stage latency is included in the planned 2026 Q3 benchmark sweep:

Configuration	What it isolates
Baseline (gateway only, no plugins)	Pipeline overhead floor
+ token validation only	Ed25519 verification cost
+ ledger write (in-memory)	Hash-chain serialization cost
+ all safety plugins	Plugin pipeline cost
+ PostgreSQL persistence	Production-write durability tail
+ Redis revocation cache	Cache hit/miss tail
+ 1-hop network (LAN)	Transport cost

The current `pnpm run bench` reports only the leftmost two configurations under in-memory persistence. The remaining configurations are scoped for Q3 2026.

7.4 Test Coverage

Table 7.4 — Test inventory (current main, commit [79d3b91](#))

Component	Tests
Core types and Zod schemas	~80
Capability tokens (issuance, delegation, revocation, chain verification)	55
Policy gateway (tier assignment, constraints, plugins, M-of-N)	256
Evidence ledger (hash chain, proof receipts, queries)	45
Bridge adapters (15 bridges)	233
OWASP ASI01-ASI10 conformance fixtures	29
Engine layer (System1, System2, HAL, capsule sandbox, avatar)	215
Persistence (in-memory, PostgreSQL, Redis)	40
Reference capsules (navigation, inspection, pick-and-place)	31
Apps (gateway-server, dashboard, sintctl, sint-mcp)	~140
Economic layer (billing, budget, route selection)	70
Avatar / CSDS	50
Integration and end-to-end	186
Edge-mode conformance	40
Conformance regression suite	~250
Total (passing)	~1,728

The OWASP conformance suite contains 29 machine-readable fixture pairs (attack vector + safe-case control) covering all 10 ASI categories. Each fixture specifies a `SintRequest`, the issued token scope, and the expected gateway decision (`allow` / `deny` / `escalate`).

7.5 OWASP Agentic Security Top 10 — Mapped Coverage

Table 7.5 — ASI mapped coverage

#	OWASP category	SINT enforcement
AS-I01	Goal Hijack / Prompt Injection	<code>GoalHijackPlugin</code> (5-layer detection)
AS-I02	Tool Misuse	Tier-based approval gates + forbidden-combo detector
AS-I03	Identity Abuse	Ed25519 capability tokens + W3C DID identity
AS-I04	Supply Chain Compromise	<code>SupplyChainVerifier</code> (model fingerprint + tool manifest)
AS-I05	Uncontrolled Code Execution	Hard T3_COMMIT classification for all <code>exec</code> / <code>bash</code> / <code>eval</code> tools
AS-I06	Memory Poisoning	<code>MemoryIntegrityChecker</code> (replay, privilege, overflow, credential funnel)
AS-I07	Inter-Agent Manipulation	Cross-agent injection detection in <code>GoalHijackPlugin</code>
AS-I08	Cascading Failure	<code>CircuitBreakerPlugin</code> + per-agent budget caps
AS-I09	Trust Exploitation	Attenuation-only delegation (max depth 3)
AS-I10	Rogue Agent	CSDS auto-escalation + CircuitBreaker auto-trip

Coverage here means *mapped coverage through conformance fixtures*: each ASI category has at least one machine-readable attack-vector fixture and one safe-case control fixture in `packages/conformance-tests/fixtures/`. Mapped coverage is not certified coverage. Microsoft Agent Governance Toolkit (2026) claims 10/10 ASI coverage for digital agents. SINT’s distinguishing position relative to that work is **physical-AI focus, actuator-bound constraints, robotics/industrial bridges, and tamper-evident evidence for safety-critical action paths** — not the count of ASI categories addressed.

7.6 Comparison with Adjacent Systems

Capability	SINT	Microsoft AGT (2026)	ROSClaw	SROS2	OPA / Rego sidecar	K8s admission controller
Runtime authorization choke point	✓	✓	—	partial (XML, startup)	✓ (digital)	✓ (control-plane)
Physical-constraint primitives	✓	—	—	—	—	—
Capability tokens with embedded constraints	✓	—	—	—	—	—
Graduated approval tiers (T0-T3)	✓	partial	—	—	—	—
Behavioral drift detection (CSDS)	✓	—	measures variance, doesn't enforce	—	—	—
Dynamic envelope from sensor state	✓	—	—	—	—	—
Tamper-evident evidence ledger	✓	—	—	—	—	—
Robotics protocol bridges (ROS 2, MAVLink)	✓	—	✓ (ROS 2 only)	✓ (ROS 2 only)	—	—
Industrial protocol bridges (OPC-UA, Sparkplug)	✓	—	—	—	—	—
Consumer / health bridges (Matter, FHIR)	✓	—	—	—	—	—
OWASP ASI mapped coverage	10/10	10/10	—	partial	—	—
Open-source license	Apache-2.0	MIT	open	open	Apache-2.0	Apache-2.0

SINT's novelty is not policy enforcement for agents in general — that is a crowded field. SINT's novelty is the *combination* of capability-bound physical constraints, bridge-level normalization across agent / robot / industrial / consumer protocols, graduated approval tiers, and hash-chained evidence — together, for actuator-facing AI systems.

7.7 Six Physical-AI Deployment Profiles

Profile 1 — Warehouse Delivery Robot (AMR). Standards: ISO 3691-4, IEC 62443.

Action	Tier	Constraint
LiDAR/camera subscribe	T0	—
Receive task via A2A	T1	—
Navigate (no humans)	T2	maxVelocity 1.5 m/s
Navigate (humans detected)	T3	maxVelocity 0.3 m/s (Δ_{human})
Gripper operation	T2	maxForce 30 N
Emergency stop	NEVER BLOCKED	I-G2

Profile 2 — Industrial Welding Arm. Standards: ISO 10218, IEC 62443.

Action	Tier	Constraint
Read joint angles	T0	—
Plan weld path	T1	—
Begin weld sequence	T2	maxForce 500 N, cell_locked: true
Move arm (human in cell)	T3	maxVelocity 0.1 m/s (ISO 10218-1 §5.4.3)
E-stop override	BLOCKED	I-G2

Profile 3 — Surgical Robot (FDA Class III). Standards: IEC 62304, IEC 60601-1-8, FDA 21 CFR Part 820, ISO 14971.

Action	Tier	Constraint
Instrument positioning	T2	maxVelocity 0.01 m/s
Force application	T2	maxForce 5 N
Electrocautery	T3	surgeon_confirmed: true (M-of-2 quorum option)
Emergency retract	NEVER BLOCKED	I-G2

Profile 4 — UAV / Drone (MAVLink). Standards: ASTM F3548-21, EU U-Space, FAA AC 107-2B.

Action	Tier	Constraint
ARM / DISARM	T3	Propulsion enable
TAKEOFF	T2	altitude \leq 120 m
MISSION_START	T3	Begins autonomous BVLOS
FENCE_DISABLE	T3	Removes safety boundary

Profile 5 — Collaborative Robot (ISO/TS 15066). Power-and-force limiting with continuous human presence: maxVelocity 0.25 m/s, maxForce 150 N (Table 1 transient contact). Per-body-region force limits: Head 130 N, Chest 140 N, Hand 140 N, Thigh 220 N.

Profile 6 — Underwater ROV (DNV-ST-0111).

Action	Tier	Constraint
Thruster control	T2	maxVelocity 1.0 m/s
Manipulator	T2	maxForce 200 N
Emergency surface	ALWAYS FORWARDED	I-G2

7.8 Compliance Crosswalk

Table 7.8a — IEC 62443 FR1-FR7 (*implementation mapping, not certified compliance*)

FR	Title	SINT mechanism
FR1	Identification & Authentication	SintCapabilityToken with Ed25519 + W3C DID portability
FR2	Use Control	Four-tier approval gate; maxRepetitions constraint; per-resource action allowlists
FR3	System Integrity	SHA-256 hash-chained ledger; ProofReceipt for T2/T3 (TEE planned)
FR4	Data Confidentiality	TLS transport; capability scope prevents sensor access without token
FR5	Restricted Data Flow	Policy Gateway allowlists; geofence constraint; bridge per-resource DFA
FR6	Timely Response	safety.estop.triggered ; I-G2 E-stop universality
FR7	Resource Availability	Per-token rate limiting; budget enforcement

Table 7.8b — EU AI Act Articles 9, 11, 12, 13, 14, 15 (*implementation support, not certified compliance*)

Article	Requirement	SINT approach
Art. 9	Risk management system	CSDS + tier system + dynamic envelopes — supporting implementation evidence
Art. 11	Technical documentation	Evidence ledger as immutable design record
Art. 12	Logging and traceability	Hash-chained ledger; tamper detection cryptographic
Art. 13	Transparency for high-risk AI	Disclosure statement + audit trail
Art. 14(4)(e)	Stop button	<code>CircuitBreakerPlugin</code> (software-level; complements certified hardware E-stop)
Art. 15	Accuracy / robustness / cybersecurity	Plugin-based runtime detection — mapped, not certified

Table 7.8c — NIST AI RMF / ISO 42001 / EU AI Act tier crosswalk

SINT tier	NIST AI RMF function	ISO/IEC 42001 clause	EU AI Act article
T0 Observe	MAP + MEASURE + MANAGE monitoring	Clause 9 + Clause 8	Art. 12 + Art. 13
T1 Prepare	GOVERN + MANAGE controlled write	Clause 8.1/8.2	Art. 9 + Art. 12
T2 Act	MANAGE risk response with oversight	Clause 8 + Clause 6	Art. 14 + Art. 15
T3 Commit	Highest-consequence GOVERN + MANAGE	Clause 8.3 + Clause 10	Art. 14(4)(e) + Arts. 9/12/15

A machine-readable crosswalk is exposed at `GET /v1/compliance/tier-crosswalk`.

7.9 Bridge Protocol Coverage

SINT bridges 15 physical, industrial, agent, and consumer protocols. Each bridge has (a) a canonical resource-URI mapper, (b) physical-context extractors, (c) interop equivalence fixtures (the same logical action expressible via two protocols produces the same normalized `SintRequest`), and (d) tier-classification defaults.

Table 7.9 — Bridge inventory

Bridge	Protocol family	Standards	Tier defaults
<code>bridge-mcp</code>	MCP tool calls	Anthropic 2024	per-tool, exec → T3
<code>bridge-ros2</code>	ROS 2 topics/services/actions	OMG DDS, ROS 2	per-topic
<code>bridge-mavlink</code>	MAVLink v2	PX4 / ArduPilot	per-command table (\$6.3)
<code>bridge-a2a</code>	Google A2A	A2A 2025	per-task
<code>bridge-grpc</code>	gRPC services	Protobuf	per-method
<code>bridge-iot</code>	Generic MQTT/CoAP	RFC 7252	sensor → T0, actuator → T2
<code>bridge-mqtt</code>	MQTT pub/sub	OASIS MQTT 5.0	per-topic
<code>bridge-mqtt-sparkplug</code>	Sparkplug B	Eclipse Sparkplug	industrial defaults
<code>bridge-opcua</code>	OPC-UA	IEC 62541	per-node, writes → T2
<code>bridge-open-rmf</code>	Open-RMF fleet	Open-RMF	per-task
<code>bridge-swarm</code>	Multi-agent	NATO STANAG 4586	collective
<code>bridge-homeassistant</code>	Home Assistant	OSS	per-device-class
<code>bridge-matter</code>	Matter	CSA 2023	per-cluster
<code>bridge-health</code>	FHIR R5 + HealthKit/Health Connect	HL7 FHIR	consent-bound
<code>bridge-economy</code>	Internal	—	per-resource

7.10 Production Readiness Signals

Beyond passing the test suite and conformance fixtures, SINT exposes several production-readiness primitives:

- `/v1/ready` orchestration health gate (PostgreSQL connectivity, Redis connectivity, revocation cache freshness, ledger hash-chain self-check on startup).
- Prometheus metrics at `/v1/metrics` covering gateway decision latency, decision rate by tier, CSDS per `foundation_model_id`, circuit-breaker state transitions, evidence-ledger write throughput, approval-queue depth.
- OpenAPI surface at `/v1/openapi.json` for SDK and integration scaffolding.
- Public protocol discovery at `/.well-known/sint.json` advertising supported bridges, tier policies, and conformance certification status.

- Docker Compose profiles for dev, edge (bandwidth-constrained T0/T1 local), prod-lite (single-node PG+Redis), Gazebo validation (simulated robotics), and Isaac Sim validation (NVIDIA simulator integration).

7.11 Planned Experiments

We list the experiments that must be executed before the evaluation can claim physical-AI safety. Each is acknowledged technical debt for this paper version.

Experiment	Result needed
ROS 2 TurtleBot / Clearpath Jackal simulation	Gateway enforcement under real ROS topics, sensor delays, actuator feedback
Gazebo / Isaac Sim closed-loop integration	Dynamic envelope behavior near obstacles and humans
MCP red-team suite (arXiv: 2601.17549 fixtures)	Prompt-injection and tool-misuse blocked; reduction toward the 12.4% post-mitigation residual reported in the source
MAVLink / PX4 adversarial test bench	Token replay, command injection, geofence bypass, mode-change cascades
PostgreSQL + Redis production-persistence benchmark	p99 latency under durable writes and revocation cache contention
Networked-gateway benchmark	Latency over 1-hop LAN; measure transport floor
Human approval-latency study	T2/T3 operational viability under realistic operator workload
Real-robot pilot (Clearpath Jackal + UR5e, Q3 2026)	At least one end-to-end actuator path under SINT control

The first results from these experiments are targeted for the IROS 2026 / IEEE RA-L submissions (Appendix E, R3).

8. Open Problems

SINT’s current implementation closes the *single-agent runtime authorization* gap for physical AI within a deployment that routes all actuator paths through the gateway. It does not close the harder problems that emerge under multi-agent, sub-human-reaction-time, federated, and economically autonomous deployments. We name five open problems here; the full research agenda — with year-by-year execution calendar, planned publication map, and key open questions — is in Appendix E.

OP-1 — Swarm authorization. A swarm of N agents, each with a valid individual capability token, can collectively execute actions (converging on a target, exceeding aggregate kinetic-energy thresholds) that no individual token captures. We need

swarm tokens with collective-constraint primitives, Byzantine-resilient k-of-N coordination, and federated trust across organizational boundaries.

OP-2 — Sub-human-reaction-time approval. Physical AI operates at 100 Hz–1 kHz; human approval has 200–500 ms median latency. A robot can execute 200–500 control cycles while waiting for approval. Open: pre-approved trajectory corridors with deviation-triggered escalation, probabilistic constraint envelopes, predictive tier assignment, and hardware-interrupt escalation paths.

OP-3 — Model-bound capability tokens. Tokens currently bind to an agent identity, not a model fingerprint. A silent model update under the same agent identity can re-pose the robot under a different effective policy. Open: tokens that carry an allowed model hash and behavioral-baseline distribution; cross-model quorum for T3; adversarial-model detection.

OP-4 — Formal verification. SINT’s invariants (I-T1 to I-G3) are enforced by runtime code and validated empirically by the conformance suite. For FDA Class III, DO-178C Level A, and IEC 62304 Class C certification, mechanized proofs are required. Open: TLA+ model-checking of I-G1/I-G2/I-G3; Coq proof of token attenuation (I-T1); Isabelle/HOL hash-chain integrity (I-G6).

OP-5 — Real-robot validation. All current benchmarks run in single-process simulation. End-to-end validation on physical robots — gateway enforcement under real ROS topics, sensor delays, actuator feedback, adversarial network conditions — is required before any safety claim graduates from “mapped coverage” to “evidence of safe operation.” A Clearpath Jackal + Universal Robots UR5e pilot is scoped for Q3 2026.

Two additional thrusts are pursued in parallel but cover ground that is more research-program than open-problem: side-channel hardening (decision-traffic normalization, sealed evidence ledger, sensor attestation) and autonomous economic coordination (ZK delivery proofs, stake-bonded tokens, MPC fleet bidding). Both are detailed in Appendix E.

9. Discussion and Limitations

We identify five technical limitations and three ethical/societal limitations of the current approach.

L1 — No formal verification yet. SINT’s invariants (§2.6, Appendix A) are enforced by runtime code, not by formal proof. Section 8 (R7) lays out the path to TLA+ / Coq / Isabelle mechanization, but as of this writing the proofs do not exist. For FDA Class III, DO-178C Level A, and ISO 26262 ASIL-D certification, this is a hard blocker; runtime testing does not substitute for formal proof in those regimes.

L2 — No real-robot evaluation. All benchmarks in §7 run on TypeScript in a simulated environment (in-memory persistence or local PostgreSQL/Redis). The 5.1 ms steady-state p99 latency is promising but has not been validated on a physical ROS 2 system with network transport, sensor delays, actuator feedback loops, and adversarial network conditions. A warehouse AMR pilot using Clearpath Jackal + Universal Robots UR5e is planned for Q3 2026; this is acknowledged technical debt.

L3 — CSDS threshold is empirically ungrounded. The escalation thresholds θ in §5.1 are design parameters informed by domain reasoning, not empirically validated values. Determining the minimum observation window and the optimal threshold per deployment context requires field data across diverse foundation models, task distributions, and operator populations. The 2026 Q4–2027 Q2 publications (IROS 2026, IEEE RA-L) target this gap.

L4 — Heuristic threat detection. `GoalHijackPlugin` and `MemoryIntegrityChecker` rely on pattern-matching heuristics (regex, keyword lists). Sophisticated adversarial prompts will evade them. A hybrid approach combining heuristic pre-filters with light-weight transformer classifiers (DistilBERT-class, sub-100 ms inference) is the planned 2027 Q3 extension.

L5 — Static tier classification. The 50+ tier-classification rules are hand-authored. A learned tier classifier trained on human-operator approval decisions could adapt to deployment-specific risk profiles and reduce false escalations. The risk: introducing a learned component into the safety path adds attack surface and complicates the certification path (L1). The 2027–2028 plan is to deploy learned classifiers in *advisory* mode first and only promote them to the safety path after they demonstrate stability over a sustained operational period.

Ethical and societal limitations.

E1 — Dual-use risk. SINT is, technically, a runtime authorization framework. Its design principles are domain-neutral. We do not include export-control restrictions, military-use carve-outs, or weapon-system blocklists in the protocol itself. The Apache-2.0 license permits use cases we would individually decline. The mitigation is community governance: SIP-001 establishes a governance process for protocol-level prohibitions, and the conformance suite includes test fixtures for weapons-targeting patterns that any conformant deployment must deny.

E2 — Civil-liberties affordances. The Health Fabric (`bridge-health`) and Consumer Smart Home (`bridge-homeassistant`) bridges process some of the most sensitive data a person produces. The design choices we have made — on-device-first processing, per-query ϵ -budget differential privacy, FHIR consent tokens with `purposeOfUse` , caregiver delegation with revocability — are deliberate civil-liberties affordances. They do not eliminate the surveillance risk inherent in connected sensors; they constrain how badly that risk can compound under standard SINT-conformant deployments.

E3 — Operator burden. Graduated approval shifts cognitive load onto human operators. If a deployment is configured to escalate frequently — too-tight CSDS θ , too many forbidden combos — operators experience alert fatigue and start *click-through approving* without due consideration. This is the well-documented failure mode of any approval-gate system (e.g., TLS warnings, IDS alerts). The mitigation surface is two-fold: (a) per-deployment θ calibration (R3) and operator-rate-of-approval limits in the dashboard; (b) UX research on operator workflows — explicitly *not* a place where we want to ship clever AI summarization shortcuts.

10. Related Work

Capability-based security. The object-capability model [Miller, Yee, Shapiro 2003] provides the theoretical foundation for SINT’s token design. Unlike ambient-authority RBAC systems, capability tokens are unforgeable, attenuatable, and revocable — properties critical for delegation in multi-agent physical deployments. Earlier capability systems (EROS, KeyKOS, Seln-Capability OS) target operating systems; SINT lifts the model to runtime authorization for embodied agents.

Safety shields. Alshiekh et al. [AAAI 2018] formalize safety shields as runtime monitors that override unsafe actions in reinforcement learning. SINT shares the interposition architecture but operates at a higher abstraction level (capability tokens and approval tiers rather than state-action shielding) and supports multi-protocol, multi-agent deployments.

Robot safety standards. IEC 62443 [IEC 2013] and ISO 10218 [ISO 2011] define security requirements for industrial control systems and robotic devices but assume deterministic, trusted controllers. SINT extends these standards to the LLM-driven case by treating the controller as untrusted and interposing a runtime authorization layer below it.

Agentic AI security. OWASP’s Agentic Security Initiative [OWASP 2026] identifies 10 risk categories for autonomous AI agents. Microsoft Agent Governance Toolkit [Microsoft 2026] addresses these for digital/software agents but does not provide physical-domain bridges, embedded physical-constraint primitives, or hash-chained evidence. SINT’s contribution is the combination of capability-bound physical constraints, multi-protocol bridge normalization, graduated approval tiers, and tamper-evident evidence for actuator-facing AI systems, with mapped coverage of the 10 ASI categories through the conformance fixture suite.

LLM-driven robotics. ROSClaw [Cardenas et al. 2026] provides the empirical foundation for SINT’s behavioral-drift detection by measuring frontier-LLM variance under matched task specifications. SayCan [Ahn et al. 2022] and Code-as-Policies [Liang et al. 2023] demonstrate LLM-to-robot pipelines but include no runtime authorization

layer. Open X-Embodiment [Padalkar et al. 2024] and OpenVLA [Kim et al. 2024] establish that cross-embodiment policy transfer is increasingly viable — strengthening the case for an embodiment-agnostic authorization layer.

ROS 2 security. SROS2 [DDS-Security 2018] secures inter-node DDS traffic via static XML policies. CCS 2022 vulnerabilities [Caiazza et al. 2022] established that this architectural choice — static policies, scattered enforcement — produces a class of access-control bypasses by construction. SINT’s single-choke-point design is a direct response.

Agent protocols. MCP [Anthropic 2024], A2A [Google 2025], ACP (IBM/Linux Foundation), AgentProtocol, ANP, and AG-UI standardize inter-agent and agent-to-tool communication. None contains a physical-constraint primitive, graduated approval flow, or hash-chained audit substrate. SINT explicitly does not compete with these protocols; it wraps them.

Evidence ledgers. Hash-chained append-only logs are well-known in the certificate-transparency literature [Laurie 2013, RFC 6962] and in the Bitcoin / blockchain space. SINT uses the same primitive at a smaller, faster scope: per-deployment ledgers with optional federated anchoring rather than global consensus.

Differential privacy. SINT’s planned DP ledger analytics (§8.5) build on Dwork & Roth [35] and the per-query ϵ -budget accounting model. The `bridge-health` package already implements per-query ϵ -budget for FHIR access.

Composable trust registries. The Open Agent Trust Registry [pshkv 2026] provides a federated registry of trusted attestation issuers with Ed25519 3-of-5 multi-signature threshold governance. The accompanying “Composable Agent Trust Stack” [arcedo 2026] documents the broader vision in which SINT, APS, and other agent-internet primitives compose.

11. Conclusion

We presented SINT Protocol, a runtime authorization framework for LLM-driven physical agents. SINT interposes a single Policy Gateway between every agent action and every actuator (under the deployment trust assumptions of §3.5), enforces six safety invariants — capability confinement (I-T1, I-T2, I-T3), graduated human oversight, behavioral drift detection, environment-adaptive constraints, active threat detection, and software-level emergency stop — and records every decision in a tamper-evident SHA-256 hash-chained ledger. In single-process, in-memory benchmarks the gateway adds 5.1 ms steady-state p99 latency, fitting within the 10 ms budget of a 100 Hz ROS 2 control loop. The reference implementation ships with approximately 1,728 tests, 15 protocol bridges, four SDKs, and mapped coverage of the OWASP Top 10 for Agent

Applications through conformance fixtures. Production-persistence benchmarks, networked-gateway benchmarks, MCP/ROS/MAVLink red-team campaigns, and real-robot validation remain to be executed (§7.11).

Our core position is that **safe physical AI requires authorization as a first-class runtime concern**, enforced cryptographically at the agent-actuator boundary — *not* as a deployment configuration, prompt-engineering convention, or post-hoc compliance audit. Alignment research focuses on what models *want*; SINT focuses on what models are *allowed to do*, *with what constraints*, *with what evidence*. The two approaches are complementary; neither suffices alone. Neither, in turn, replaces hardware-rated emergency-stop controllers or domain-rated safety standards.

We close with five open problems (§8) — swarm authorization, sub-human-reaction-time approval, model-bound capability tokens, formal verification, and real-robot validation — and a five-year research agenda in Appendix E. SINT is open source under Apache-2.0 at github.com/sint-ai/sint-protocol; the conformance suite, fixtures, and benchmarks are reproducible from the same repository.

Acknowledgments

The author thanks the ROSClaw team for the empirical foundation on inter-model behavioral variance; the OWASP Agentic Security Initiative for the threat taxonomy that structures SINT’s conformance suite; the Open Agent Trust Registry collaborators (Arcede, Insumer, QNTM, AgentID, Agora) for ongoing federation experiments; and the broader open-source contributors to the sint-protocol monorepo.

AI Disclosure Statement

This paper was drafted with the assistance of Claude (Anthropic; model `claude-sonnet-4-6`, 1M context) acting on materials authored by the listed human author and resident in the public repository github.com/sint-ai/sint-protocol. The assistant performed: (a) extraction and synthesis from the in-repo whitepaper, SPAI 2026 submission, research agenda, README, and source code; (b) structural drafting of sections under explicit human direction; (c) bilingual abstract production. The human author retains substantive authorship, has reviewed every claim against the underlying source code and test suite, and is solely responsible for the paper’s contributions, accuracy, and any errors. No proprietary or third-party data not present in the public repository was used. The full conversation transcript and prompt log are retained by the author and available for editorial review upon request, in line with venue-specific AI disclosure requirements.

Funding Statement

This work received no external funding. SINT Protocol development is supported by SINT AI Lab / PSHKV Inc. as an open-source initiative under the Apache-2.0 license. The author declares no competing financial interests beyond direct employment at PSHKV Inc., the steward of the open-source project.

Author Contribution Statement (CRediT)

Illia Pashkov — Conceptualization (lead), Methodology (lead), Software (lead), Investigation (lead), Writing — original draft (lead), Writing — review and editing (lead), Visualization (lead), Project administration (lead), Funding acquisition (lead).

Data Availability Statement

All software, configuration, conformance fixtures, benchmark scripts, and documentation referenced in this paper are publicly available at github.com/sint-ai/sint-protocol under the Apache-2.0 license. Benchmark results are reproducible via `pnpm install && pnpm run build && pnpm run bench` on a system meeting the listed prerequisites (Node.js 22+, pnpm 9+). The OWASP ASI conformance fixtures are at [packages/conformance-tests/fixtures/](https://packages.conformance-tests/fixtures/). No human-subject data, proprietary datasets, or restricted resources were used.

Ethics Declaration

This paper presents a software framework with no human-subject experiments. The framework’s design includes explicit civil-liberties affordances (on-device-first health data processing, per-query ϵ -budget differential privacy, FHIR consent tokens with revocability) discussed in §9. The dual-use considerations are discussed in §9 (E1); SIP-001 establishes a governance process for protocol-level prohibitions of harmful use cases.

Limitations Section (Summary)

Detailed in §9: L1 no formal verification yet; L2 no real-robot evaluation (Q3 2026 pilot planned); L3 CSDS thresholds empirically ungrounded; L4 heuristic threat detection; L5 hand-authored tier rules; E1 dual-use risk; E2 civil-liberties affordances are not eliminations; E3 operator-burden / alert-fatigue risk.

References

Citation format: IEEE. Persistent identifiers (DOI, URL, repository hash) are included where available. Standards bodies cited inline. All references are reachable via the listed identifier as of 2026-05-25.

- [1] I. S. Cardenas, M. A. Arnett, N. C. Yeo, L. Sah, and J.-H. Kim, “ROSClaw: An OpenClaw ROS 2 framework for agentic robot control and interaction,” *arXiv:2603.26997*, 2026. [Online]. Available: <https://arxiv.org/abs/2603.26997>
- [2] “MCP Security Analysis: Architectural vulnerabilities in the Model Context Protocol,” *arXiv:2601.17549*, 2026. [Online]. Available: <https://arxiv.org/abs/2601.17549>
- [3] OWASP Foundation, “OWASP Top 10 for Agentic Applications (Agentic Security Initiative),” 2026. [Online]. Available: <https://owasp.org/>
- [4] Anthropic, “Model Context Protocol Specification,” 2024. [Online]. Available: <https://spec.modelcontextprotocol.io/>
- [5] Google, “Agent-to-Agent (A2A) Protocol,” 2025. [Online]. Available: <https://github.com/google/A2A>
- [6] Microsoft, “Agent Governance Toolkit,” 2026. [Online]. Available: <https://github.com/microsoft/agent-governance-toolkit>
- [7] M. Alshiekh, R. Bloem, R. Ehlers, B. Könighofer, S. Niekum, and U. Topcu, “Safe reinforcement learning via shielding,” in *Proc. AAAI*, 2018, pp. 2669–2678.
- [8] M. S. Miller, K.-P. Yee, and J. Shapiro, “Capability myths demolished,” Combex Inc., Tech. Rep., 2003.
- [9] M. Ahn et al., “Do as I can, not as I say: Grounding language in robotic affordances (SayCan),” in *Proc. CoRL*, 2022.
- [10] J. Liang et al., “Code as policies: Language model programs for embodied control,” in *Proc. ICRA*, 2023.
- [11] A. Padalkar et al., “Open X-Embodiment: Robotic learning datasets and RT-X models,” in *Proc. ICRA*, 2024.
- [12] M. J. Kim et al., “OpenVLA: An open-source vision-language-action model,” *arXiv:2406.09246*, 2024.
- [13] G. Caiazza et al., “Security analysis of ROS 2 and SROS2,” in *Proc. ACM CCS*, 2022.
- [14] IEC 62443-3-3:2013, “Industrial communication networks — IT security for networks and systems — Part 3-3: System security requirements and security levels,” International Electrotechnical Commission, 2013.

- [15] ISO 10218-1:2011 and ISO 10218-2:2011, “Robots and robotic devices — Safety requirements for industrial robots,” International Organization for Standardization.
- [16] ISO/TS 15066:2016, “Robots and robotic devices — Collaborative robots,” International Organization for Standardization.
- [17] ISO 14971:2019, “Medical devices — Application of risk management to medical devices,” International Organization for Standardization.
- [18] IEC 62304:2015, “Medical device software — Software life cycle processes,” International Electrotechnical Commission.
- [19] IEC 60601-1-8:2020, “Medical electrical equipment — Part 1-8: General requirements — Alarm systems,” International Electrotechnical Commission.
- [20] U.S. Food and Drug Administration, “21 CFR Part 820 — Quality System Regulation,” 2024.
- [21] NIST, “Artificial Intelligence Risk Management Framework (AI RMF 1.0),” NIST AI 100-1, 2023. doi:10.6028/NIST.AI.100-1.
- [22] NIST, “Guide to Operational Technology (OT) Security,” NIST SP 800-82 Rev. 3, 2023.
- [23] European Parliament and Council, “Regulation (EU) 2024/1689 — Harmonised rules on artificial intelligence (EU AI Act),” 2024.
- [24] ISO/IEC 42001:2023, “Information technology — Artificial intelligence — Management system,” International Organization for Standardization.
- [25] NATO STANAG 4586, “Standard Interfaces of UAV Control System (UCS) for NATO UAV Interoperability.”
- [26] ASTM F3548-21, “Standard Specification for UAS Remote ID Network Publishing and Discoverability.”
- [27] FAA, “Advisory Circular AC 107-2B — Small Unmanned Aircraft Systems,” 2021.
- [28] DNV-ST-0111, “Assessment of station keeping capability of dynamic positioning vessels,” DNV.
- [29] IEC 62541-2017, “OPC Unified Architecture,” International Electrotechnical Commission.
- [30] Eclipse Foundation, “Sparkplug B Specification 3.0,” 2022.
- [31] Connectivity Standards Alliance, “Matter Specification 1.3,” 2023.
- [32] HL7 International, “FHIR R5 — Fast Healthcare Interoperability Resources,” 2023.

- [33] B. Laurie, “Certificate Transparency,” *Commun. ACM*, vol. 57, no. 10, pp. 40–46, 2014.
 - [34] B. Laurie, A. Langley, and E. Kasper, “Certificate Transparency,” IETF RFC 6962, 2013.
 - [35] C. Dwork and A. Roth, “The algorithmic foundations of differential privacy,” *Found. Trends Theor. Comput. Sci.*, vol. 9, no. 3–4, pp. 211–407, 2014.
 - [36] D. Boneh, B. Lynn, and H. Shacham, “Short signatures from the Weil pairing,” in *Proc. ASIACRYPT*, 2001, pp. 514–532.
 - [37] Trusted Computing Group, “TPM 2.0 Library Specification,” TCG, 2023.
 - [38] Trusted Computing Group, “DICE — Device Identifier Composition Engine,” TCG, 2022.
 - [39] W3C, “Decentralized Identifiers (DIDs) v1.0 — Core architecture, data model, and representations,” W3C Recommendation, 2022.
 - [40] I. Pashkov et al., “SINT Protocol — Reference implementation,” 2026. [Online]. Available: <https://github.com/sint-ai/sint-protocol>
 - [41] I. Pashkov, “SINT Protocol: Runtime Authorization as a Safety Shield for LLM-Driven Physical Agents,” IJCAI/ECAI 2026 Workshop on Safe Physical AI (SPAI 2026), submission, May 2026.
 - [42] I. Pashkov et al., “Open Agent Trust Registry,” 2026. [Online]. Available: <https://github.com/pshkv/open-agent-trust-registry>
 - [43] G. Klein et al., “seL4: Formal verification of an OS kernel,” in *Proc. ACM SOSP*, 2009.
 - [44] L. Lamport, *Specifying Systems: The TLA+ Language and Tools for Hardware and Software Engineers*. Addison-Wesley, 2002.
 - [45] G. Necula and P. Lee, “Safe kernel extensions without run-time checking,” in *Proc. OSDI*, 1996.
-

Appendix A — Formal Invariants and DFA

A.1 Invariant Table

ID	Name	Statement
I-T1	Token Attenuation	For any delegation chain $t_0 \rightarrow t_1 \rightarrow \dots \rightarrow t_k$, $\text{scope}(t_i) \subseteq \text{scope}(t_{i-1})$ for all $i \in [1, k]$. Equivalently, $\text{actions}(t_i) \subseteq \text{actions}(t_{i-1}) \wedge \text{constraints}(t_i) \supseteq \text{constraints}(t_{i-1})$ (constraints only tighten) $\wedge \text{expiresAt}(t_i) \leq \text{expiresAt}(t_{i-1})$.
I-T2	Token Unforgeability	A valid <code>CapabilityToken</code> carries a valid Ed25519 signature over its canonical serialization by <code>issuer</code> . Forging a valid token without access to <code>issuer.privateKey</code> is computationally infeasible under the Ed25519 unforgeability assumption.
I-T3	Physical Constraint Primacy	If <code>token.constraints.maxVelocityMps = v</code> , then no downstream layer (bridge, plugin, transform) may issue a <code>PolicyDecision</code> allowing an action with effective velocity $> v$. Effective velocity may be reduced by <code>DynamicEnvelopePlugin</code> .
I-G1	No Bypass	Physical actuation is reachable only from DFA state <code>ACTING</code> (or <code>COMMITTING</code> for T3). <code>ACTING</code> and <code>COMMITTING</code> are reachable only via the transition <code>POLICY_EVAL → PLANNING → {OBSERVING, PREPARING, ESCALATING}</code> . The transition into <code>POLICY_EVAL</code> requires a valid <code>tokenId</code> whose token verifies under I-T1, I-T2, I-T3.
I-G2	E-stop Universality	For any non-terminal DFA state s , the event <code>estop</code> transitions $s \rightarrow \text{ROLLED-BACK}$ unconditionally. <code>estop</code> is <i>not</i> a request flowing through <code>intercept()</code> ; it is a sideband signal.
I-G3	Ledger Primacy	The transition <code>COMMITTING → COMPLETED</code> requires a <code>ledger_committed</code> event with valid <code>previousHash</code> / <code>eventHash</code> linkage. No action is recorded as completed without a corresponding hash-chained ledger event.
I-G4	Tier Monotonicity	Tier assignment is monotonic across the request lifecycle. If $\text{Tier}(r, t_0) = T_2$, then $\text{Tier}(r, t_1) \geq T_2$ for $t_1 > t_0$. Tier may escalate; it never demotes.
I-G5	Ledger Append-Only	Ledger events are insert-only. No <code>UPDATE</code> or <code>DELETE</code> is permitted on the events table. Schema-level constraints enforce.
I-G6	Hash Chain Integrity	$\ell_k.\text{previousHash} = \text{SHA256}(\text{canonical}(\ell_{k-1}))$ for all $k \geq 1$. Genesis: $\ell_0.\text{previousHash} = "0" \times 64$.
I-G7	Budget Fail-Deny	Insufficient <code>EconomyPlugin</code> balance results in <code>PolicyDecision.action = "deny"</code> , never <code>"escalate"</code> . Operators cannot approve their way out of a depleted budget.
I-G8	Timeout Fail-Deny	If an approval flow exceeds <code>timeoutMs</code> , the default <code>fallbackAction</code> is <code>"deny"</code> . A deployment may set <code>fallbackAction: "allow"</code> for non-safety paths only, with documented rationale.

A.2 DFA State Table

State	Description	Exits
IDLE	Pre-request	<code>submit</code> → PENDING
PENDING	Request received, awaiting schema validation	<code>schema_ok</code> → POLICY_EVAL , <code>schema_fail</code> → FAILED
POLICY_EVAL	Token + scope + rate + constraint + plugins running	<code>tier_assigned</code> → PLANNING , <code>denied</code> → FAILED
PLANNING	Tier-based path selection	<code>tier_check</code> → OBSERVING v PREPARING v ESCALATING
OBSERVING	T0 path: read-only logged	<code>done</code> → COMPLETED
PREPARING	T1 path: low-impact audited	<code>done</code> → COMPLETED
ESCALATING	T2/T3 path: awaiting approval	<code>approved</code> → ACTING (T2) v COMMITTING (T3), <code>denied</code> → FAILED , <code>timeout</code> → FAILED
ACTING	T2 execution	<code>done</code> → COMPLETED , <code>error</code> → ROLLEDBACK
COMMITTING	T3 irreversible execution + ledger commit	<code>ledger_committed</code> → COMPLETED , <code>error</code> → ROLLEDBACK
COMPLETED	Terminal: success	—
FAILED	Terminal: denied or invalid	—
ROLLEDBACK	Terminal: E-stop or rollback	—

A.3 Proof Sketches

The following sketches are not mechanically verified. They are informal arguments under the deployment trust assumptions of §3.5 and the cryptographic assumptions of §2.3. Mechanizing them is Open Problem 4 (§8) and Appendix E thrust 7.

Sketch for I-G1 (No Bypass). Let τ be the DFA transition relation. By inspection of the state table, the only incoming transitions to **ACTING** are **ESCALATING** $\text{--approved(T2)} \rightarrow$ **ACTING** and to **COMMITTING** are **ESCALATING** $\text{--approved(T3)} \rightarrow$ **COMMITTING**. **ESCALATING** has a single incoming transition: **PLANNING** $\text{--tier_check(T} \geq 2) \rightarrow$ **ESCALATING**. **PLANNING** has a single incoming transition: **POLICY_EVAL** $\text{--tier_assigned} \rightarrow$ **PLANNING**. **POLICY_EVAL** has a single incoming transition: **PENDING** $\text{--schema_ok} \rightarrow$ **POLICY_EVAL**. **PENDING** has a single incoming transition: **IDLE** $\text{--submit} \rightarrow$ **PENDING**. The transition **PENDING** \rightarrow **POLICY_EVAL** requires `SchemaValidator` success; **POLICY_EVAL** \rightarrow **PLANNING** requires `TokenValidator`, `ResourceScopeChecker`, `PhysicalConstraintChecker`, and the plugin pipeline to return non-deny. Therefore, under the deployment trust assumption that all actuator-facing paths route through `intercept()` (§3.5), no request reaches **ACTING** or **COMMITTING** without token validation under I-T2 and policy evaluation. \square

Sketch for I-G2 (E-stop Universality). The `estop` event is implemented as a side-channel callback registered with the gateway by an operator with a circuit-breaker capability. It is not flowed through `intercept()`. By construction of the DFA implementation, every non-terminal state (`IDLE`, `PENDING`, `POLICY_EVAL`, `PLANNING`, `OBSERVING`, `PREPARING`, `ESCALATING`, `ACTING`, `COMMITTING`) handles the `estop` event by transitioning to `ROLLEDBACK` and emitting an `estop_received` ledger event. Operationally, `estop` is delivered by direct method invocation on the gateway instance and is not subject to plugin ordering. \square

Sketch for I-T1 (Token Attenuation). By induction on chain depth `k`. Base case: `t_0` (root) trivially satisfies `scope(t_0) \subseteq scope(t_0)`. Inductive step: assume `scope(t_{k-1}) \subseteq scope(t_{k-2})`. The `TokenValidator.attenuates(parent, child)` check at delegation time verifies (i) `actions(child) \subseteq actions(parent)` by set-membership, (ii) `constraints(child) \geq constraints(parent)` by per-field numeric/geometric tightening checks (a smaller numeric ceiling, a polygon contained within the parent polygon, an earlier expiry), and (iii) signature chain validity by Ed25519 verification. If `attenuates(t_{k-1}, t_k)` returns true, then `scope(t_k) \subseteq scope(t_{k-1}) \subseteq scope(t_{k-2})` by transitivity. \square

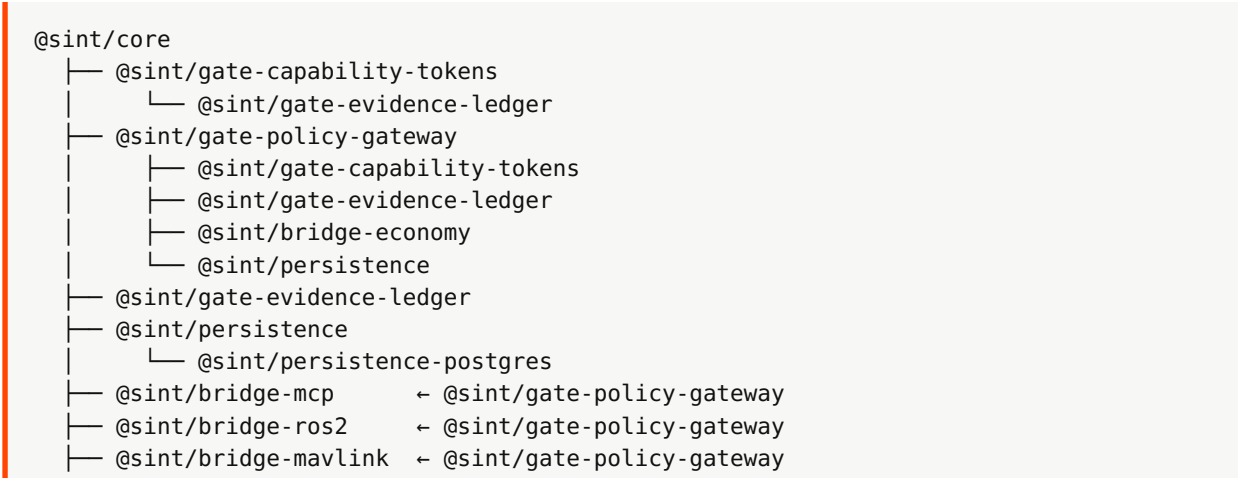
Sketch for I-G6 (Hash Chain Integrity). Suppose `l_k` is modified to `l'_k`. Then `eventHash(l'_k) \neq eventHash(l_k)` with probability `1 - 2^{-128}` under the SHA-256 collision-resistance assumption (birthday bound). For all `j > k`, the entry `l_j` carries `previousHash = eventHash(l_{j-1})`, which was computed against the original chain. Recomputing `eventHash(l'_k)` requires also recomputing `previousHash(l_{k+1}) = eventHash(l'_k)`, which in turn requires recomputing `eventHash(l_{k+1}) = SHA256(canonical(previousHash || serialized(l_{k+1})))`, propagating through `l_{k+2}`, ..., `l_n`. Therefore, tamper detection at any single event implies inconsistency at all subsequent events. \square

These sketches do not constitute mechanized proof. The current commitment (Appendix E thrust 7) is to encode I-G1, I-G2, I-G6 in TLA+; I-T1 in Coq; and to attempt Isabelle/HOL mechanization of hash-chain integrity under standard cryptographic assumptions.

Appendix B — Environment Variable Reference

Variable	Default	Description
SINT_ENV	development	development production
SINT_STORE	memory	memory postgres redis
SINT_CACHE	memory	memory redis
DATABASE_URL	—	PostgreSQL connection string
REDIS_URL	—	Redis connection string
SINT_API_KEY	—	Required for production HTTP API access
SINT_REQUIRE_SIGNATURES	false	Enforce per-request Ed25519 signature verification
SINT_WS_ALLOW_QUERY_API_KEY	true	Allow API key in WebSocket query string (set false in prod)
SINT_TOKEN_TTL	86400000	Default token TTL (ms)
SINT_MAX_DELEGATION_DEPTH	3	Maximum delegation-chain depth
SINT_CS_DS_THRESHOLD	0.30	Default CS_DS escalation threshold (alias retained: SINT_CSML_THRESHOLD for v1.0 backward-compat)
SINT_CIRCUIT_BREAKER_THRESHOLD	5	Consecutive denials before auto-trip
SINT_APPROVAL_TIMEOUT	30000	Default approval timeout (ms), max 300_000
SINT_LOG_LEVEL	info	debug info warn error

Appendix C — Package Dependency Graph




```

├─ @sint/bridge-a2a      ← @sint/gate-policy-gateway
├─ @sint/bridge-grpc     ← @sint/gate-policy-gateway
├─ @sint/bridge-iot      ← @sint/gate-policy-gateway
├─ @sint/bridge-mqtt-sparkplug ← @sint/gate-policy-gateway
├─ @sint/bridge-opcua    ← @sint/gate-policy-gateway
├─ @sint/bridge-open-rmf ← @sint/gate-policy-gateway
├─ @sint/bridge-swarm    ← @sint/gate-policy-gateway
├─ @sint/bridge-homeassistant ← @sint/gate-policy-gateway
├─ @sint/bridge-matter   ← @sint/gate-policy-gateway
├─ @sint/bridge-health   ← @sint/gate-policy-gateway
├─ @sint/bridge-economy
├─ @sint/engine-system1
├─ @sint/engine-system2
├─ @sint/engine-hal
├─ @sint/engine-capsule-sandbox
├─ @sint/avatar
├─ @sint/client
├─ @sint/sdk
├─ @sint/conformance-tests ← all of the above

```

Appendix D — Compliance Mapping (Consolidated)

D.1 — OWASP ASI ↔ SINT plugin/mechanism

ASI	Title	Primary mechanism	Secondary
ASI01	Prompt Injection	<code>GoalHijackPlugin</code> (5-layer)	T3 hard-classification of exec
ASI02	Tool Misuse	Tier system	Forbidden-combo detector
ASI03	Identity Abuse	Ed25519 + W3C DID	Token registry
ASI04	Supply Chain	<code>SupplyChainVerifier</code>	Bridge integrity manifest
ASI05	Code Execution	Hard T3_COMMIT	<code>ArgInjectionDetector</code>
ASI06	Memory Poisoning	<code>MemoryIntegrityChecker</code>	Cross-session continuity probe
ASI07	Inter-Agent Manipulation	Cross-agent injection check	A2A scope validation
ASI08	Cascading Failure	<code>CircuitBreakerPlugin</code>	Per-agent budget cap
ASI09	Trust Exploitation	Attenuation-only delegation	Max depth 3
ASI10	Rogue Agent	CSDS auto-escalation	CircuitBreaker auto-trip

D.2 — EU AI Act Article ↔ SINT artifact

Article	Requirement	SINT artifact
Art. 9	Risk management	CSDS + DynamicEnvelope
Art. 11	Technical docs	Evidence ledger as design record
Art. 12	Logging	Hash-chained ledger
Art. 13	Transparency	Disclosure statement + ledger API
Art. 14(4)(e)	Stop button	CircuitBreakerPlugin
Art. 15	Robustness / cybersecurity	Plugin runtime detection

D.3 — IEC 62443 FR ↔ SINT mechanism

(See §7.8 Table 7.8a.)

D.4 — NIST AI RMF function ↔ SINT capability

RMF function	SINT capability
MAP	Resource taxonomy + tier classification + bridge inventory
MEASURE	CSDS + p99 latency + conformance suite + ASI fixtures
MANAGE	Tier escalation + CircuitBreaker + approval flow + budget
GOVERN	SIP governance + audit ledger + conformance certification

Appendix E — Research Agenda 2026–2031

This appendix contains the detailed research agenda formerly presented as §8 in v1.0. The agenda is forward-looking and exploratory; it is not part of the protocol’s normative core. Components listed here are at maturity levels *Planned* or *Speculative* per §3.6.

Calendar note. This document is dated 2026-05-25 — late Q2 2026. Items marked completed (✓) in the calendar below cover work through 2026-05-25 only. Items in the same quarter that have not yet shipped on the canonical *main* branch are marked (~) for in-flight and (planned) for unstarted. Future-quarter items have no marker.

E.1 Research Thrust 1 — Swarm Coordination Security (2026 Q3 → 2027 Q4)

Threat. A drone swarm of N agents shares a task. Each individual agent has a valid capability token. But the *collective* action — N drones converging on a target point, N forklifts cross-aisle handoffs in a warehouse, N surgical instruments operating in a shared cavity — is dangerous in ways no individual token expresses.

Current SINT gap. The capability token is per-agent. The `SwarmCoordinator` plugin (§5.8) enforces a small fixed set of collective constraints (total kinetic energy, minimum inter-agent distance, max T2+ concurrency, max escalated fraction). It does not handle: Byzantine-compromised swarm members, federated trust across organizations, swarm-CSDS over heterogeneous backends, or formal proofs of collective-safety properties.

Research directions.

- **R1.1 — Swarm token.** A group capability token encoding collective constraints (max swarm density agents/m³, max collective kinetic energy $\Sigma \frac{1}{2}mv^2$, min inter-agent distance, synchronization window). Open question: how is the swarm token attenuated as it propagates? Does each member receive an attenuated child, or do all members share a single swarm token verified by quorum?
- **R1.2 — Collective CSDS.** Extend `computeCsml()` to agent cohorts. Tighter threshold θ_{swarm} than per-agent θ . Open: does swarm CSDS compose linearly over members, or does it require a non-linear coupling term capturing emergent behavior?
- **R1.3 — Byzantine-resilient coordination.** Threshold signature scheme: collective actions require k-of-N agents to present valid tokens. Extends the existing M-of-N operator quorum to N distributed *agents*. Builds on Boneh-Lynn-Shacham BLS signatures for aggregation.
- **R1.4 — Federated swarm trust.** Swarms crossing organizational boundaries (Company A's drones receiving handoff from Company B's logistics) need a federated capability negotiation protocol. The Open Agent Trust Registry [see §8.4] is the proposed root of trust.

Target standards. NATO STANAG 4586 (UAS Control System interfaces), MIL-STD-1553B (avionics bus security), ASTM F3411-22a (UAS remote ID broadcast format).

Planned outputs. Draft `@sint/bridge-swarm` v2 with swarm-token primitive (Q3 2026); IROS 2026 submission with collective-CSDS empirical validation (August 2026 deadline); IEEE RA-L journal extension (Q4 2026); ICRA 2027 swarm Byzantine-resilience paper (Q1 2027 deadline).

E.2 Research Thrust 2 — Sub-Human-Reaction-Time Safety (2026 Q4 → 2028 Q2)

Threat. Physical AI operates at 1 kHz (ROS 2 control loops at 100 Hz–1 kHz). Human approval has 200–500 ms latency. A robot can execute 200–500 control cycles while waiting for human approval — during which the physical state is undefined. Today operators pre-approve trajectories; but the robot's real-time sensor state diverges from the planned state. At what divergence threshold should the pre-approved plan be invalidated?

Current SINT gap. Static `recentActions[]` and `physicalContext` enable forbidden-combo detection and Δ human escalation at request granularity. SINT does not currently express *probabilistic* constraints, *pre-approved corridors* (trajectory envelopes), or *predictive* tier assignment.

Research directions.

- **R2.1 — Probabilistic constraint envelopes.** Replace binary constraint checks with Gaussian confidence bounds.

```
typescript interface StochasticConstraint { nominalValue: number; stdDev: number;
confidenceLevel: 0.95 | 0.99 | 0.999; // P(safe) }
```

A velocity of 0.5 ± 0.1 m/s is allowed at 99% confidence; 0.5 ± 0.5 m/s escalates.

- **R2.2 — Incremental corridor approval.** Instead of approving discrete actions, approve *corridors*: the gateway approves a 10-second trajectory envelope. Actions within the envelope auto-allow. Deviations beyond a per-axis tolerance require real-time escalation.
- **R2.3 — Predictive tier assignment.** An ML model trained on `decisionLedger` data predicts which upcoming actions will need T2/T3 approval, pre-fetching human attention so the human is *already looking at the queue* when the action arrives.
- **R2.4 — Hardware interrupt escalation.** Bypass the software stack for emergency tier promotion. `safety.force.exceeded` triggers a hardware interrupt that sets a capability token flag before any software can react.

Target standards. ISO 26262 (functional safety for automotive, ASIL-D analog for robotics), IEC 61508 SIL 3, ISO 13849 PL e.

Planned outputs. Probabilistic constraint envelope spec (RFC) Q4 2026; trajectory-corridor reference capsule Q1 2027; IEEE TASE submission (CSDS empirical validation including corridor mode) Q2 2027; ASIL-equivalent industrial safety paper to IEEE TIE Q1 2028.

E.3 Research Thrust 3 — LLM Behavioral Drift and Model Fingerprint Binding (2026 Q3 → 2029 Q2)

Threat. ROSClaw [arXiv:2603.26997] documented a $3.4\times$ behavioral divergence between frontier LLMs on identical robotic tasks. A robot certified safe under GPT-4o is *not* certified safe under any other foundation model. Yet today nothing binds a capability token to a specific model; a silent update can re-pose the robot under different effective policy.

Current SINT gap. CSDS detects *historical* divergence. It cannot predict future divergence and cannot catch a newly deployed model before it causes harm.

Research directions.

- **R3.1 — Model fingerprint tokens.** Capability tokens bound to a specific model hash, not just an agent identity. Token invalidated when the model updates.

```
typescript interface ModelBoundToken extends CapabilityToken { modelConstraints:
{ allowedModelIds: string[]; // e.g. ["gpt-4o-2024-11-20"] maxModelVersion?:
string; // semver ceiling modelFingerprintHash?: string; // SHA-256 of weights/con-
fig }; }
```

- **R3.2 — Behavioral baseline tokens.** Issue a token encoding an expected behavior *distribution* (action frequency histogram, velocity distribution, force profile). CSDS compares live behavior against the baseline distribution; divergence triggers escalation.
- **R3.3 — Cross-model quorum for T3.** Irreversible actions require sign-off from $k \geq 2$ distinct foundation models. If GPT-5 and Claude 5 both recommend the action, confidence is higher than any single model.
- **R3.4 — Adversarial-model detection.** Detect a model that *appears* aligned but produces sequences consistent with a known attack pattern when integrated over a horizon — the “sleeper agent” failure mode.

Target standards. NIST AI RMF MEASURE-2.7 (robustness under adversarial conditions); EU AI Act Art. 15 cybersecurity-of-AI requirements.

Planned outputs. ModelBoundToken Schema 1.0 (Q3 2026); cross-model quorum spec (RFC) Q4 2026; Nature Machine Intelligence submission “Behavioral fingerprints as a primary safety primitive for embodied LLMs” Q2 2027; ICML 2027 paper on adversarial-model detection.

E.4 Research Thrust 4 — IoT and Edge-Mesh Authorization (2027 Q1 → 2029 Q4)

Threat. A factory has 10,000 IoT sensors, 500 robots, and 50 AI agents. The centralized SINT gateway is a bottleneck. Every sensor read going through `PolicyGateway.intercept()` is unrealistic. The question: how do you push SINT’s security model down to devices that have 64 KB of RAM and 100 Hz duty cycles?

Current SINT gap. The reference gateway is implemented in TypeScript on Node 22; capability tokens are 384+ bytes; Ed25519 verification requires $\sim 150 \mu\text{s}$ on an M3. None of this fits a Cortex-M0 microcontroller.

Research directions.

- **R4.1 — SINT-nano lightweight tokens.** Strip Ed25519 to the minimum viable representation: 32 B subject + 32 B issuer + 8 B expiry + 4 B resource hash + 64 B signature = **140 bytes**. Compatible with constrained C-language clients.

- **R4.2 — Offline token verification.** Edge devices verify tokens locally using cached public keys. No network round-trip for T0/T1. Only T2/T3 require gateway contact, with a sub-second deadline before falling back to deny.
- **R4.3 — Hierarchical trust domains.** Factory floor runs a local SINT proxy that handles T0/T1 autonomously. Only T2+ propagate to the central gateway. The evidence ledger is replicated asynchronously to a regional aggregator, then to the global ledger root, with cryptographic anchoring at each layer.
- **R4.4 — MQTT/CoAP bridge maturity (`bridge-iot`).** Bridge SINT into the IoT protocol stack. MQTT topic = SINT resource URI; CoAP `PUT /sensor/temperature` → T0 (auto-allow, no gateway); CoAP `PUT /actuator/valve` → T2 (requires gateway check). 56 tests already exist in `bridge-iot` v0.10; the research thrust extends this to constrained-resource targets.

Target standards. IEC 62443-3-3 (System security requirements); NIST SP 800-82 (OT security); IETF CoAP RFC 7252; IETF COSE RFC 8152 (compact crypto for IoT).

Planned outputs. SINT-nano token specification (RFC) Q1 2027; embedded Rust SDK Q2 2027; IEEE IoT Journal submission Q3 2027; USENIX Security 2028 paper on hierarchical trust domains.

E.5 Research Thrust 5 — Side-Channel Hardening (2027 Q3 → 2030 Q2)

Threat. An attacker who cannot compromise the cryptographic layer can still observe the *pattern* of SINT gate decisions — which commands are approved at what tier, with what latency — to infer the robot’s mission, route, and physical state. This is the physical analog of traffic analysis.

Current SINT gap. Decision-traffic patterns are unmasked; ledger analytics (`/v1/ledger`) expose per-agent trajectories; CSDS scores are computed and published in aggregate without privacy treatment.

Research directions.

- **R5.1 — Decision-traffic normalization.** Pad T0/T1 decisions to uniform timing (adds artificial latency to prevent timing side channels). Quantify the latency budget cost.
- **R5.2 — Differential privacy for ledger analytics.** CSDS computations run on the evidence ledger. Publishing aggregate CSDS scores must not leak individual trajectory data. Apply per-query ϵ -budget DP noise to public metrics. `bridge-health` already implements per-query ϵ -budget for FHIR access; the same primitive extends to operator-facing analytics.
- **R5.3 — Sealed evidence ledger.** For T3 events, the event payload is encrypted under a TEE-sealed key. Only the operator’s HSM (PKCS#11) can decrypt post-hoc. The hash chain remains public for integrity; content is private.

- **R5.4 — Sensor attestation.** Each sensor produces signed readings (extending TPM remote attestation patterns to robotics); the gateway verifies sensor integrity before honoring `physicalContext.currentVelocityMps` or `humanDetected` as escalation triggers. A compromised sensor that *lies* about human presence is a critical adversarial pattern.

Target standards. ISO/IEC 19790 (FIPS 140-3 equivalent for crypto modules); TCG TPM 2.0 and TCG DICE for sensor attestation; IEEE 802.1AE (MACsec) for transport.

Planned outputs. Decision-traffic normalization measurement study Q4 2027; DP ledger analytics implementation Q2 2028; sealed evidence ledger spec Q3 2028; sensor attestation profile Q4 2028; IEEE Security & Privacy 2030 paper.

E.6 Research Thrust 6 — Autonomous Economic Coordination (2028 Q1 → 2031 Q2)

Threat. Drone fleets bid on delivery tasks in real-time auctions. A robot accepts a task, performs it, and expects payment — all autonomously, without human intermediation. The economic layer must be as secure as the physical layer. Today’s SINT has metered billing and trust-tier pricing; it does not have on-chain economic settlement, ZK delivery proofs, or multi-party computation for fleet bidding.

Current SINT gap. `@sint/bridge-economy` implements per-agent budget, trust-tier pricing, and cost-aware route selection. It does not yet do cryptographic stakes, zero-knowledge proofs of physical delivery, or MPC-based bid coordination.

Research directions.

- **R6.1 — Zero-knowledge delivery proofs.** A drone proves it reached waypoint W at time T without revealing its full trajectory (privacy-preserving delivery verification). ZK-SNARK over GPS + barometer + timestamp inputs.
- **R6.2 — Capability token as economic instrument.** Tokens carry a *stake* amount. If the token holder causes a safety violation, the stake is slashed on-chain. The token *is* the bond. Extends the existing `sla.bond.slashed` ledger event to Solana / Ethereum economic settlement.
- **R6.3 — Multi-party computation for fleet bidding.** N drones bid on a task without revealing their individual capabilities to competitors. MPC produces a winning assignment; each drone receives an attenuated token for its specific subtask.
- **R6.4 — Cross-chain settlement.** Existing `bridge-ethereum`, `bridge-solana`, and `bridge-nano` packages provide the rails; the research is in protocol-level guarantees around atomicity and rollback when economic settlement diverges from physical outcome.

Target standards. Emerging W3C Verifiable Credentials for Payment; FATF Travel Rule for autonomous-agent settlement (open question).

Planned outputs. ZK delivery proof spec Q1 2028; stake-bonded token implementation Q3 2028; MPC fleet-bidding paper to ACM CCS 2029; Nature Communications submission on robot-Solana staking 2030.

E.7 Research Thrust 7 — Formal Verification of Gateway Invariants (2028 Q2 → 2031 Q4)

Threat. SINT’s invariants (I-T1 through I-G3) are currently enforced by runtime code, validated empirically by 1,728 tests. For FDA Class III certification, aerospace DO-178C Level A, and IEC 62304 Class C, formal proofs are required.

Current SINT gap. No formal proofs exist. The gateway DFA (12 states, 18 transitions) is documented in Markdown; the invariants are stated informally; no model checker has verified them.

Research directions.

- **R7.1 — TLA+ specification.** Model-check core gateway invariants. Specifically: I-G1 (no bypass) reduces to a reachability property on the DFA — `ACTING` is reachable only via the `POLICY_EVAL → PLANNING` transition. I-G2 (E-stop universality) is a fairness property. I-G3 (ledger primacy) is a sequencing property.
- **R7.2 — Coq/Lean proofs.** Formally verify token delegation attenuation (I-T1). The proof obligation: `attenuates(parent, child) ∧ valid(parent) ∧ valid(child) ⇒ scope(child) ⊆ scope(parent)`.
- **R7.3 — Isabelle/HOL hash-chain proofs.** Prove that any modification of an event `e_k` invalidates all events `e_{k+1..n}` under the assumption that SHA-256 is collision-resistant.
- **R7.4 — Certification path.** Use the formal proofs to support FDA 510(k) submissions for surgical-robotics deployments and DO-178C Level A certification for drone autonomy.

Target standards. FDA 21 CFR Part 820 (medical-device QMS) with IEC 62304 Class C; DO-178C Level A (avionics software); ISO 26262 ASIL-D (automotive).

Planned outputs. TLA+ specification of the gateway DFA Q3 2028; Coq attenuation proof Q1 2029; ITP 2029 conference submission; FDA 510(k) pre-submission with formal-methods package 2030; IEEE TDSC special-issue submission 2031.

E.8 Year-by-Year Execution Calendar

```

=== 2026 ===
Q1 (✓): Avatar Layer + CSDS auto-escalation; MAVLink bridge; APS↔SINT interop
Q2 (~): SwarmCoordinator v1; MQTT/CoAP bridge (bridge-iot); ISO 10218 body-force model
Q3:    Phase 9 – token-registry, SafetyPermitPlugin, IotInterceptor (56 tests),
      /v1/registry routes [PLANNED – not yet shipped as of 2026-05-25]
Q4:    ModelBoundToken Schema 1.0; cross-model quorum spec (RFC);
      IROS 2026 submission; IEEE RA-L empirical validation;

```



```
npm publish 8 @sint/ packages; Constraint Language CL-1.0
```

```
=== 2027 ===
```

- Q1: SINT-nano lightweight tokens (140-byte format); offline T0/T1 verification;
OPC-UA bridge hardening (IEC 62541 conformance);
embedded Rust SDK MVP; trajectory-corridor reference capsule
- Q2: Hardware Security Module integration (PKCS#11);
hierarchical trust domain proxies; cross-model quorum implementation;
Nature Machine Intelligence submission (R3.2)
- Q3: Hybrid heuristic + transformer threat-detection classifier;
ICML 2027 submission (R3.4 adversarial-model detection);
decision-traffic normalization measurement study
- Q4: Federated cross-org capability negotiation;
Open Agent Trust Registry as PKI root;
USENIX Security 2028 paper drafting (R4.3)

```
=== 2028 ===
```

- Q1: ZK delivery proof spec (R6.1); probabilistic envelope production rollout;
FDA 510(k) pre-submission scoping
- Q2: Differential privacy for ledger analytics (R5.2);
TLA+ specification of gateway DFA (R7.1);
IEEE TASE submission (R2 corridor mode)
- Q3: Sealed evidence ledger spec (R5.3); stake-bonded token impl (R6.2);
sensor attestation profile (R5.4)
- Q4: Sensor attestation production rollout;
IEEE TIE submission (R2 ASIL-equivalent industrial safety paper)

```
=== 2029 ===
```

- Q1: Coq attenuation proof complete (R7.2); MPC fleet bidding (R6.3);
ACM CCS 2029 submission
- Q2: IROS 2029 swarm formal verification paper;
IEEE Security & Privacy 2030 submission drafting (R5)
- Q3: Cross-chain settlement guarantees (R6.4); first robot-Solana staking pilot
- Q4: ITP 2029 formal-methods presentation;
FDA 510(k) submission with formal-methods package

```
=== 2030 ===
```

- Q1: IEEE Security & Privacy 2030 conference presentation;
Isabelle/HOL hash-chain proofs (R7.3)
- Q2: Sleeper-agent / adversarial-model detection deployment
- Q3: Aerospace DO-178C Level A certification pathway scoping
- Q4: Nature Communications "Robot-Solana staking" submission

```
=== 2031 ===
```

- Q1: End-to-end formal-methods package for FDA Class III surgical robotics
- Q2: IEEE TDSC special issue: "Runtime authorization for embodied LLMs – 5 years"
- Q3: ISO/IEC standards-track submission proposal
- Q4: Retrospective: open-source post-mortem and 2032–2036 horizon

E.9 Planned Publications Map

Year	Venue	Topic	Mapped thrust
2026	IROS 2026 (Aug)	Collective CSDS empirical validation	R1
2026	IEEE RA-L	CSDS inter-model variance measurement	R3
2027	Nature Machine Intelligence	Behavioral fingerprints for embodied LLMs	R3
2027	ICML 2027	Adversarial-model detection	R3
2027	IEEE IoT Journal	SINT-nano edge authorization	R4
2027	IEEE TASE	Probabilistic constraint corridors	R2
2028	USENIX Security 2028	Hierarchical trust domains	R4
2028	IEEE TIE	ASIL-equivalent industrial safety	R2
2029	ACM CCS 2029	MPC for fleet bidding	R6
2029	ITP 2029	Coq attenuation proofs	R7
2030	IEEE S&P 2030	Side-channel hardening for SINT	R5
2030	Nature Communications	Stake-bonded capability tokens	R6
2031	IEEE TDSC	5-year retrospective	All

E.10 Key Open Questions

The following questions structure the agenda above and remain genuinely open as of the date of this revision:

- 1. Is there a formal proof of the DFA invariants I-G1 / I-G2 / I-G3?** Hypothesis: TLA+ model-checking can mechanize I-G1 (reachability) and I-G2 (universality); I-G3 (sequencing) requires a temporal-logic specification.
- 2. What is the minimum viable CSDS window for real-time escalation?** Hypothesis: 50 events sufficient for warning; 200 events stable. Needs empirical validation across model backends and task distributions.
- 3. Can APS attestation grade be used as a prior for CSDS θ ?** Grade 0 agents $\rightarrow \theta=0.10$ (untrusted); Grade 3 $\rightarrow \theta=0.40$ (legal entity).
- 4. What is the attack surface of a MAVLink-SINT deployment?** Known: token replay (mitigated by `timeWindow`), command injection (mitigated by physical-constraint check). Unknown: timing attacks on the intercept path; bridge-mapping confusion across PX4 / ArduPilot / DroneCAN variants.
- 5. How does CSDS behave under coordinated adversarial prompt injection (arXiv:2601.17549)?** Hypothesis: prompt injection increases AR (denial rate) and decreases CR (completion rate). CSDS should detect injection campaigns as CSDS $> \theta$ within 20–30 events.

6. **What composability does swarm-CSDS obey?** Linear composition over members, or does it require non-linear coupling terms? Empirical question with simulation precedence.
 7. **Is a 140-byte SINT-nano token verifiable on a Cortex-M0 in under 5 ms?** Engineering question with measurable answer; success enables the whole IoT thrust (R4).
 8. **Can ZK delivery proofs be generated within the flight-time budget of a quadcopter?** Open: SNARK proving time vs. battery time.
 9. **What is the right legal model for an autonomous economic agent that stakes capital and may be slashed?** Open jurisdictional question — likely 2028–2030 policy work.
 10. **Where is the line between “safety shield” and “alignment substitute”?** The position of this paper is that they are *complementary*: aligned models can still produce out-of-policy actions under distribution shift; SINT does not replace alignment, but it does substantially compress the consequence of alignment failures into recoverable runtime denials.
-
-

End of paper.

Repository: github.com/sint-ai/sint-protocol — commit [79d3b91](#), 2026-05-25.

Suggested citation: Pashkov, I. (2026). *SINT Protocol: Runtime Authorization and Evidence Logging for LLM-Driven Physical AI Systems*. Working paper, revision 1.1, SINT AI Lab / PSHKV Inc. Available at: <https://github.com/sint-ai/sint-protocol>