Inside LSAS

How deterministic validation, policy packs, escalation, and audit telemetry work

A TECHNICAL NARRATIVE FOR PRODUCT, PLATFORM, SECURITY, AND COMPLIANCE STAKEHOLDERS.

Audience: executive sponsors, product leaders, platform/security teams, legal/compliance stakeholders.

LSAS Stack by Inference Stack LLC
Enterprise AI Execution Authority™

ADDRESS
Inference Stack LLC
5365 W. 73rd Pl.
Arvada, CO 80003

CONTACT
+1 720.334.3838
www.lsas-stack.com
www.lsas-stack.com/contact

Inside LSAS

What this brief is for

This document explains LSAS as a runtime system: what sits where, what happens to each request, and what proof is emitted after a decision.

It is intentionally implementation-aware and operational. The goal is to accelerate architecture, security, and design review conversations.

What LSAS is — and what it is not

What LSAS is

- An application-layer governance runtime between products and execution targets
- A deterministic decisioning layer for high-stakes AI traffic
- A place where policy packs, validators, evidence checks, and escalation paths are applied inline
- A source of structured telemetry and decision proof

What LSAS is not

- Not a generic chatbot wrapper
- Not a substitute for legal judgment, clinical judgment, or internal compliance programs
- Not a certification by itself
- Not merely a logging utility after the fact

Core runtime pipeline

The current LSAS product narrative describes a six-step pipeline. That sequence matters because it keeps control logic explicit, understandable, and auditable. The system is not treating the request as an opaque blob. It is progressively determining what kind of interaction this is, what claims or actions are present, what evidence exists, what policy thresholds apply, and what the resulting decision should be.

1. Classify risk

Identify whether the request enters a legal, clinical, financial, security, or customer-impacting workflow.

2. Inspect claims

Detect assertions, recommendations, citations, or actions that require stronger controls.

3. Check evidence

Assess grounding in approved sources, retrieval results, structured data, or deterministic validation.

4. Enforce thresholds

Apply policy rules for evidence sufficiency, redaction, validators, and escalation triggers.

5. Decide at runtime

Allow, constrain, redact, abstain, block, or escalate based on risk and proof sufficiency.

6. Emit proof

Return a decision envelope with findings, remediations, policy version, and audit metadata.

Inside LSAS

The core control surfaces

Control surface	Role in the runtime	Why it matters
Policy packs	Versioned rule bundles that define thresholding, detectors, validators, redaction rules, and escalation behavior.	They make governance explicit and testable rather than tribal or ad hoc.
Deterministic validators	Checks against structured rules, known patterns, or approved schemas.	They reduce ambiguity in areas where the enterprise cannot accept model-only judgment.
Evidence thresholds	Requirements for approved sources, retrieval support, structured data, or review before trust is granted.	They prevent unsupported claims from surviving into high-stakes output.
Decision envelopes	Structured results that capture outcome, risk tier, findings, remediations, and audit metadata.	They give multiple teams a shared artifact to inspect.
Governed overrides	Explicit exception paths with authority, confirmation, and audit capture.	They allow flexibility without dissolving the policy boundary.
Telemetry and timelines	Derived signals, event trails, change history, and workflow timeline records.	They support incident review, control drift detection, and executive visibility.

Decision outcomes

The LSAS site describes deterministic outcomes such as ALLOW, REDACTED, BLOCKED, and ESCALATE_HITL. Those statuses are more than display labels. They are operating states that downstream systems and reviewers can act on.

Outcome	What it means operationally
ALLOW	The output or action met threshold under the current policy tier and can continue without intervention.
REDACTED / CONSTRAINED	The request can proceed only after controlled transformation such as identifier removal, output narrowing, or policy-safe rewriting.
BLOCKED	The request or release path cannot continue under current conditions.
ABSTAIN	The system withholds an answer or action because it cannot meet policy requirements with sufficient proof.
ESCALATE_HITL	A human decision path is required before release or execution.

Inside LSAS

Representative decision envelope

Field	Representative value
workflow_domain	legal / clinical / fintech / security / accessibility
policy_tier	low / standard / regulated / locked_down
decision	ALLOW / REDACTED / BLOCKED / ESCALATE_HITL
risk_signals	claim risk, data sensitivity, actionability, boundary crossing
findings	rule hits, unsupported citations, identifier exposure, secret leakage, etc.
evidence_status	approved sources present / validator pass-fail / missing support
remediation	redact, request grounded retry, escalate reviewer, block release
audit_metadata	tenant, app, environment, actor, timestamp, policy-pack version

Why the envelope matters

A decision envelope gives product, security, compliance, and leadership one artifact to reason about. That reduces the usual fragmentation where prompt logs, application logs, and governance commentary all live in different places.

Deployment and boundary posture

- Private deployment by default, including private single-tenant, customer-hosted cloud/VPC, or self-hosted/on-prem options
- Derived-only telemetry by default; raw prompts and completions can be disabled entirely
- Segmentation by tenant, app, and environment to maintain clear blast-radius boundaries
- Operational hooks that can integrate into incident management, SIEM, or other event sinks
- Governed healthcare connector-boundary patterns for FHIR/OAuth search and read flows in the current sandbox scope

Inside LSAS

Where sandbox behavior is useful

The LSAS sandbox should be read as a controlled evaluation environment rather than a claim of universal workflow coverage. That is still valuable. A serious sandbox should show baseline behavior, governed behavior, escalation mechanics, and audit evidence under representative scenarios. The current LSAS sandbox does exactly that by exposing guided scenarios, policy tiers, override simulation, and decision metadata in a reviewable interface.

Implementation implications

- LSAS works best when governance is attached to named workflows and boundaries, not as a generic site-wide toggle
- Policy ownership must be explicit: who owns the baseline pack, who can approve exceptions, and who decides evidence sufficiency for each high-stakes domain
- Telemetry should be wired into ordinary operational review, not reserved for audit season
- Runtime control should be paired with rollout discipline: baseline, governed comparison, exception design, and scale plan

Selected references

1. LSAS Stack website — home. Application-layer runtime positioning, pipeline, risk domains, telemetry, and deployment options. https://www.lsas-stack.com/

2. LSAS Stack — Sandbox / Guided demo. Scenario list, decision envelope behavior, governed overrides, and audit metadata framing. https://www.lsas-stack.com/playground

3. LSAS Stack — Compliance posture. Controls mapping language, deployment/residency posture, data minimization, auditability, and operational hooks. https://www.lsas-stack.com/compliance-posture

4. LSAS Stack — Privacy. Derived-only telemetry, raw prompt/completion defaults, and customer-boundary deployment posture. https://www.lsas-stack.com/privacy

5. LSAS Stack — Pricing / implementation approach. Implementation phases, inputs, first-90-day cadence, executive telemetry, and delivery outputs. https://www.lsas-stack.com/pricing

6. Inference Stack — Contact. Inference Stack contact page; based in Denver, Colorado; serving clients worldwide. https://www.inference-stack.com/contact