Jimini: AI Policy Firewall

- Inline enforcement
- auditability
- safe rollout

Problem

- AI agents can leak secrets, PII, and regulated content
- Manual review is slow and non-auditable.

What Jimini Does

- Evaluates every inbound/outbound message against YAML rules (regex, length, optional LLM, direction, endpoint)
- Returns block / flag / allow with deterministic precedence.

Architecture (Concept)

- Agent/App → Jimini → External
- Components: FastAPI gateway, rules loader (hot reload), enforcement engine, hash-chained audit log, metrics + SARIF, optional webhook + OTEL spans, optional OpenAI check.

Rules-as-Code

- pattern + min_count
- max_chars
- llm_prompt (fail-safe)
- applies_to
- endpoints (exact/prefix/glob)
- shadow_override
- suppression of generic API-1.0 when specific secret matches.

Shadow → Enforce Rollout

- 1) Shadow baseline 2) Enforce high-confidence secrets 3) Expand to regulated packs 4) SIEM dashboards (SARIF) 5) Ticketing integration (future).

Evidence & Observability

- /v1/metrics
- /v1/audit/verify
- /v1/audit/sarif
- Webhooks on block
- OTEL spans (optional)
- Hash chain integrity.

Security & Noise Reduction

- Precedence block>flag>allow
- Specific secret suppression
- Fail-safe LLM path
- Minimal surface (single POST)
- Deterministic outputs.

Value Summary

- Risk reduction
- Compliance evidence
- Fast integration (one POST)
- No model retraining
- Quantifiable metrics for governance.

Pilot Success Metrics

- Violations/1K msgs
- False positive rate <5% (enforced)
- Time-to-detect (real-time)
- Secret exposure reduction vs baseline.

Risks & Mitigations

- Over-blocking: shadow first
- Rule sprawl: lint + packs
- False positives: scoping/suppression
- Missing patterns: metrics review.

Ask / Next Steps

- Approve 2-week shadow pilot (≤1 engineer day)
- Deliver baseline report + enforce list, then phased rollout.

One-Liner

- Jimini = seatbelt + black box for AI agent I/O: rules-as-code guardrails, safe rollout, immutable evidence.