Sentinel Architecture Freeze (v1)

Purpose
This document freezes the architecture of the Sentinel Linux SSH Security Monitoring system. It defines clear responsibilities for each layer and identifies legacy logic that must be revamped or reduced as the system matures.

Final Pipeline

Detection → Scoring → Correlation → Decision → Execution Planning → (Execution – Disabled)

Layer Responsibilities

- **detector/**: Detects raw patterns and emits factual alerts only. Must not infer attack types, severity, or responses.
- **alerting/alert_score.py**: Converts alert intensity into numeric score and severity labels only.
- **correlation/**: Aggregates alerts by IP, infers attack_type (FAST_BRUTE, SLOW_BRUTE, PASSWORD_SPRAY), and computes incident timelines.
- **response/decision_engine.py**: Maps incidents to policy decisions such as MONITOR, FLAG_FOR_REVIEW, or BLOCK_SIMULATED.
- **response/execution_planner.py**: Produces safe, auditable execution plans in DRY-RUN mode.
- **execution/**: Dormant execution capability. Present for SOAR completeness but disabled by design.
- **state.py**: Maintains system memory including deduplication and cooldown tracking.

Required Revamps & Reductions

- Detectors must remove attack_type, severity, response logic, and execution-related code.
- Execution engine must not auto-trigger actions and should remain guarded behind LIVE_MODE.
- state.py must be reduced to memory-only responsibilities and stripped of execution decisions.
- Duplicate computations (severity, attack classification) must exist in exactly one layer.
- Legacy debug or execution-plan prints should remain disabled or behind explicit debug flags.

Design Principle

Lower layers describe reality. Upper layers assign meaning and intent. Only the topmost layer may interact with the operating system, and only when explicitly enabled.

Status
Architecture frozen. Future work should focus on refinement, deployment safety, and documentation rather than structural redesign.