Sovereign Sanctuary Elite - System Architecture

Version: 2.0.0 **Date:** 2026-01-25 **Classification:** Internal Technical Documentation

---

1. Executive Summary

The Sovereign Sanctuary Elite system is a distributed, self-healing automation platform designed for zero-drift, deterministic deployments with cryptographic traceability. This document provides a comprehensive overview of the system architecture, component interactions, and operational principles.

2. Architectural Principles

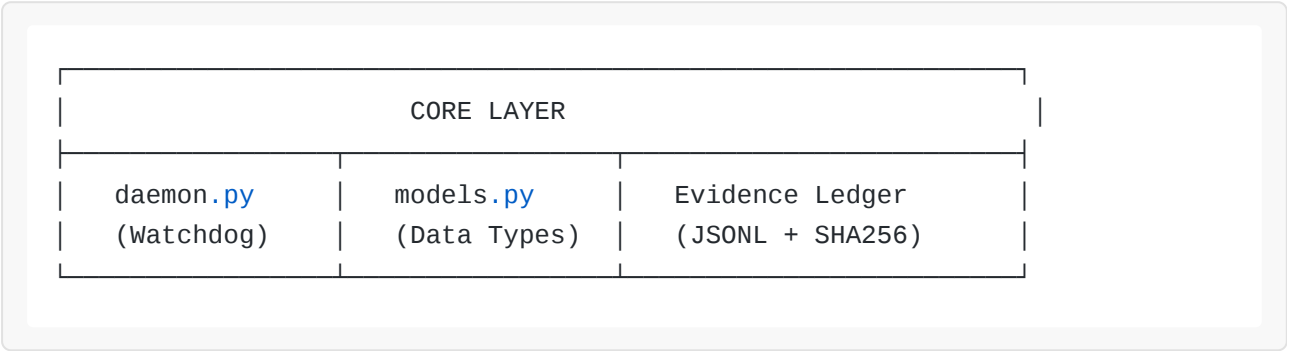
The system is built on five core architectural principles:

Principle	Description
Zero-Dependency Mindset	All components are designed to operate without external SaaS dependencies
Cryptographic Integrity	Every action produces verifiable, tamper-evident artifacts
Evidence-Bound Execution	Decisions are traceable through an immutable audit ledger
Self-Healing Resilience	Autonomous recovery from failures without human intervention
Deterministic Deployments	Identical inputs always produce identical outputs

3. System Components

3.1 Core Layer

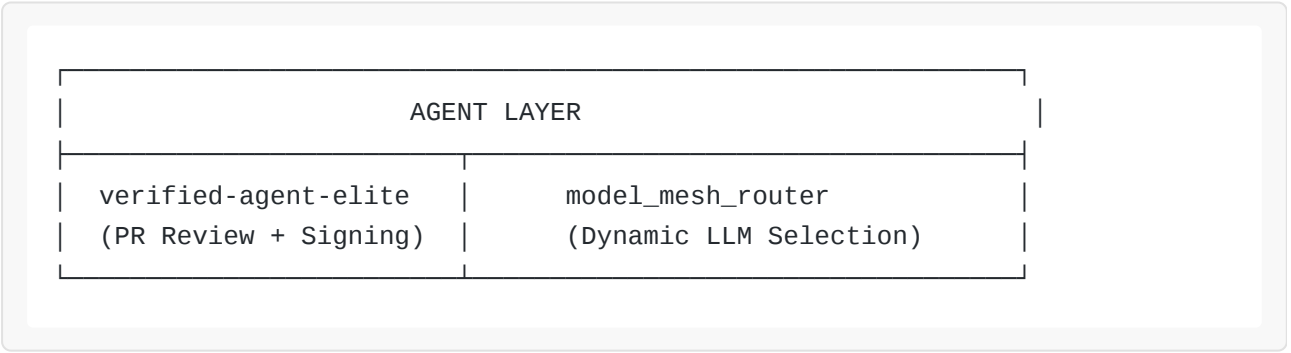
The core layer provides foundational services for all other components:



- **daemon.py**: The main watchdog process that orchestrates all system activities
- **models.py**: Pydantic data models ensuring type safety across the system
- **Evidence Ledger**: Append-only log with cryptographic hashes for audit trails

3.2 Agent Layer

The agent layer contains AI-powered components that perform automated tasks:



- **verified-agent-elite.ts**: Cryptographically signed PR review agent
- **model_mesh_router.py**: Intelligent routing to optimal LLM based on task requirements

3.3 Tools Layer

Supporting utilities for system operations:

```
┌──────────────────────────────────────────────────────────────┐
│                        TOOLS LAYER                           │
├──────────────────────────────┬───────────────────────────────┤
│  self_heal_monitor.py        │   flight_control_daemon.py    │
│  (Health + Recovery)         │   (File Watch + Seal)         │
├──────────────────────────────┼───────────────────────────────┤
│  push_system.py              │   seal_file.py                │
│  (Git + Webhook Push)        │   (SHA256 Sealing)            │
└──────────────────────────────┴───────────────────────────────┘
```

4. Data Flow Architecture

4.1 Evidence Flow

```
User Action → Agent Processing → Ledger Entry → SHA256 Seal → SITREP Update
     |              |                |              |              |
     ▼              ▼                ▼              ▼              ▼
  [INPUT]      [TRANSFORM]       [RECORD]       [VERIFY]      [REPORT]
```

4.2 Self-Healing Flow

```
Health Check → Threshold Breach → Heal Strategy → Learn Event → SITREP
Update
     |              |                |              |              |
     ▼              ▼                ▼              ▼              ▼
  [MONITOR]      [DETECT]         [RECOVER]      [LEARN]       [REPORT]
```

5. Node Topology

The system operates across a distributed node topology connected via Tailscale:

Node	Role	IP (Tailscale)	Responsibilities
PC1 (Blade)	Primary	100.94.217.81	Main compute, agent execution
PC2 (Echo)	Compute	100.94.217.82	Secondary compute, failover
PC4 (Local)	Controller	127.0.0.1	Orchestration, monitoring
NAS	Storage	192.168.1.x	Evidence archive, backups

6. Security Model

6.1 Cryptographic Chain

All outputs are signed using HMAC-SHA256:

```
Input Hash (SHA256) → Agent Processing → Output + Audit Trail → HMAC Signature
```

6.2 Verification Gate

Before any agent output is accepted, it must pass the verification gate:

1. Confidence score ≥ 0.9
2. No ESCALATE action
3. Risk score < 75
4. All findings have evidence
5. No CRITICAL severity findings

7. Configuration Schema

The system is configured via `swarm_config.json`:

```
{
  "nodes": { ... },
  "thresholds": {
    "cpu_max": 95,
    "memory_max": 90,
    "disk_min": 10
  },
  "model_mesh": {
    "default_model": "gpt-4.1-mini",
    "routing_strategy": "cost_optimized"
  }
}
```

8. Operational States

The system operates in one of four states:

State	Description	SITREP Color
GREEN	All systems nominal	✅
YELLOW	Warning threshold breached	⚠
RED	Critical threshold breached	🚨
BLACK	System offline or unreachable	⬛

9. Integration Points

9.1 External APIs

- **OpenAI API**: LLM inference for agents
- **GitHub API**: Repository operations via `gh` CLI
- **Tailscale API**: Node discovery and health

9.2 Internal Interfaces

- **Evidence Ledger**: JSONL append-only log
- **SITREP Board**: Markdown status document
- **Learn DB**: Pattern recognition database

Document Control

Version	Date	Author	Changes
2.0.0	2026-01-25	Manus AI	Initial architecture document