ZERO-DEPENDENCY PYTHON

3 VERIFIED UPSTREAM BUGFIX PATCHES

Sentinel Minion

Verifiable Proof-of-Inference (PoI) Audit & Deterministic Secret Redaction Guard for the **Cortensor** decentralized inference network, built on OpenMinion 0.1 patterns.

41 / 41

`pytest -v` passing (`tests/test_sentinel_minion.py`)

3 Patches

Local verified OpenMinion & installer diffs + reproducers

Pre + Post

Deterministic secret redaction before miner egress & JSONL export

SHA-256

Byzantine-resilient N-node PoI consensus & `poi_digest`

LIVE WEB INSPECTOR

`https://alphonshhhomenicmtu.github.io/sentinel-minion-cortensor/`

GITHUB REPOSITORY

`https://github.com/alphonshhhomenicmtu/sentinel-minion-cortensor`

DEMO VIDEO (1080P)

`https://www.youtube.com/watch?v=uKIHzoATPuA`

Decentralized Inference Needs Egress Redaction & Quorum Verification

Sending raw agent prompts to untrusted miner nodes without redaction or consensus exposes secrets and accepts silent hallucinations.

Risks in Naive Decentralized Agent Pipelines

- **Credential Leakage to Miners:** Prompts and tool outputs often contain JSON-quoted API keys, `ghp_` / `hf_` / `xoxb-` tokens, or base64 `Bearer` headers.
- **Single-Miner Hallucination & Byzantine Drift:** Trusting one inference node gives zero protection against corrupted weights or stale outputs.
- **Unbounded Latency & Cost Spikes:** No session-level SLA enforcement for `p95_latency_ms` or `total_cost_usd`.
- **Fragile Upstream Tooling:** Real bugs in OpenMinion's upgrader, installer `pkill` race, and redaction regexes caused silent failures.

Sentinel Minion Security & Audit Guarantees

- **Pre-Flight & Post-Flight Scrubbing:** Redacts secrets before prompts leave the host and again before writing JSONL audit dossiers.
- **Multi-Node PoI Quorum:** Queries `N` Cortensor miners, normalizes text, isolates Byzantine outliers, and emits a SHA-256 `poi_digest`.
- **Hard SLA Telemetry Gate:** Evaluates per-node and session latency/cost budgets against explicit thresholds.
- **3 Reproducible Upstream Patches:** Root-caused, patched, and regression-tested directly against OpenMinion 0.1 patterns.

Four-Stage Verifiable Inference Pipeline (`src/sentinel_minion/`)

Every module is pure Python with zero external runtime dependencies for deterministic execution and auditability.

1. `redaction.py`

- Scrubs JSON-quoted `api_key` fields
- Matches `ghp_`, `github_pat_`, `hf_`, `xox*-`, `sk-`
- Handles base64/OAuth2 `Bearer` (`+/=`)

→

2. `poi.py`

- Queries `N` Cortensor miner nodes
- Normalizes outputs & computes quorum
- Flags outliers & emits SHA-256 `poi_digest`

→

3. `telemetry.py`

- Checks `p95_latency_ms <= max_latency_ms`
- Enforces `total_cost_usd <= max_cost_usd`
- Tracks quorum availability SLA

→

4. `minion.py`

- Runs post-flight secret redaction
- Writes tamper-evident JSONL dossier
- Ready for OpenMinion handoff/replay

Audit Dossier Guarantees

- **Deterministic Digest:** Identical normalized quorum responses always produce the exact same SHA-256 `poi_digest` across machines.
- **Zero Secret Persistence:** Even if a Cortensor miner echoes sensitive text, post-flight redaction scrubs it prior to disk persistence.

Three Local Upstream Bugfix Patches (`upstream_patches/`)

Each bug was isolated locally with a standalone reproducer and patch diff ready for upstream PR submission (not yet merged upstream).

PATCH ID	ROOT CAUSE IDENTIFIED	REPRODUCTION ARTIFACT	FIX & STATUS
Patch 0001	Upgrade script cross-user `$HOME` mismatch & unchecked `cp` silently failed or overwrote wrong paths.	`repro_01_upgrade_home_mismatch.sh`	VERIFIED Resolved target user home explicitly & added strict copy checks.
Patch 0002	Installer clobbered existing `.env` secrets and triggered a `pkill` self-kill race during restart.	`repro_02_env_overwrite_pkill.sh`	VERIFIED Preserved existing `.env` safely & scoped PID termination.
Patch 0003	OpenMinion redaction missed JSON-quoted keys, `github_pat_` / `hf_` / `xox*-` prefixes, and base64 `Bearer` tails (`+/=`).	`test_repro_03_openminion_redaction.py`	VERIFIED Hardened regexes & added unit test coverage.

41/41 Pytest Pass & Deterministic Local PoI Simulator

Full local adversarial test coverage paired with an interactive static GitHub Pages inspector for judges.

Pytest Verification (`uv run --with pytest pytest -v`)

41 / 41 PASSED (0.04s)

- Validates pre/post-flight redaction across multi-word JSON keys, numeric secrets, GitHub/HF/`xox[a-z]-`/OpenAI tokens, and OAuth2 Bearer tails.
- Verifies Sybil duplicate-ID rejection, tie-split rejection, Byzantine outlier isolation, and tamper-evident SHA-256 `poi_digest` receipts.
- Confirms `RouterFailoverClient` circuit breakers, bounded `429`/`5xx` backoff, and `NaN`/`Inf` SLA rejection.

Live Static Web Inspector (`web/index.html`)

- **Secret Redaction Tester:** Paste raw prompts/JSON payloads in-browser using the same `xox[a-z]-` pattern as Python.
- **PoI Quorum Simulator (LOCAL FIXTURE):** Hardcoded 4-miner local fixture demonstrating outlier isolation & SHA-256 digests (not a live Cortensor receipt).
- **SLA Telemetry & Patch Cards:** Latency/cost threshold checks plus all 3 local reproducer patch cards (`upstream_patches/`).
- Deployed on GitHub Pages with zero build step and zero console errors.

Scaling Sentinel Minion Across the Cortensor Network

Ready for immediate OpenMinion 0.1 adoption and decentralized inference SLA enforcement.

Phase 1 · Upstream Merge

- Submit Patches `0001–0003` directly to OpenMinion upstream.
- Package `sentinel-minion` as a drop-in pre/post-flight middleware plugin.

Phase 2 · Live Miner Routing

- Connect N-node PoI quorum directly to live Cortensor router endpoints.
- Weight miner selection dynamically by historical quorum agreement & p95 latency.

Phase 3 · On-Chain Attestation

- Anchor periodic SHA-256 `poi_digest` Merkle roots on-chain for verifiable inference SLA proofs.
- Export CI/CD compliance bundles from JSONL audit dossiers.

LIVE WEB INSPECTOR

`https://alphonshhhomenicmtu.github.io/sentinel-minion-cortensor/`

GITHUB REPOSITORY

`https://github.com/alphonshhhomenicmtu/sentinel-minion-cortensor`

DEMO VIDEO

`https://www.youtube.com/watch?v=uKIHzoATPuA`