

TECHNICAL ARCHITECTURE // v1.0 // CONFIDENTIAL

SENTINEL

How the Platform Is Built

The engineering blueprint behind the Track-Back Defense Platform: four core modules, a tiered air-gapped vault, a predictive threat-modeling layer, and the chain-of-crime-scenes evidence model — every component designed to observe, preserve, and report, and never to reach into anyone else's systems.

COLLECT-AND-REPORT ONLY

SYNTHETIC DATA · NO LIVE MALWARE

LOCAL-FIRST

Prepared for the founder · Brandon, Florida · Confidential

System Overview

SENTINEL is built as a set of small, independent services that each run locally and talk to one another through plain files and a local database — not through the open internet. This "local-first" design is deliberate: it keeps sensitive data on the operator's own hardware, makes the whole platform air-gappable, and means a single family, a business, or a government can run exactly the same code at different scales. Every prototype in this document is real, runnable Python that was built and tested — not a mock-up.

The guiding rule at every layer is the same one that governs the whole company: SENTINEL only ever operates on its own infrastructure. It collects evidence of what attackers do to the decoys and reports that evidence to law enforcement. It never reaches into, damages, or steals from anyone else's systems, never executes captured malware, and uses only synthetic, clearly-labelled demo data.

Deception Layer

HONEYPOT + AUTO-CLONE + QUARANTINE

Decoy servers and bait files that catch intruders, clone new decoys on demand, and quarantine anything suspicious.

Analysis Layer

FORENSICS MICROSCOPE + FRAGMENT CATALOG

Fingerprints captured samples by reading bytes only, and catalogs the fragments criminals leave behind.

Evidence Layer

CHAIN OF CUSTODY + RANSOM TRACE

Hashes and packages everything into court-ready reports, and follows the public money trail to law enforcement.

Protection Layer

VAULT + FILTER GATEWAY + PACKET LAYER

Air-gapped tokenized storage, a link/attachment detonation filter, and a jurisdiction-aware egress overlay.

Common technical foundation

Every service is a small Python program. The interactive pieces use Flask and bind only to 127.0.0.1 (the local machine) so nothing is exposed to the network by default. State is stored in local SQLite databases and in plain outbox/ folders for the evidence packages. There is no cloud dependency, no telemetry, and no third-party account required to run the platform.

Honeypot & Deception Grid

The honeypot is a Flask decoy server that presents a convincing but clearly-labelled fake admin login and a fake "data center" file portal. It looks vulnerable; it is pure bait. Every interaction — timestamp, source IP (honoring X-Forwarded-For), user-agent, method, path, any posted credentials, and all headers — is written to a SQLite events table. Common intrusion probes (`/.env`, `/wp-login.php`, `/.git/config`, `/api/v1/keys`) are trapped and logged.

Components

- » `honeypot_server.py` — the decoy web server and logging engine, bound to 127.0.0.1:8080.
- » `canary_token.py` — generates watermarked bait files, each embedding a unique token and a callback URL; the beacon endpoint records a track-back event when the bait is opened on the attacker's own machine.
- » `db.py` — shared SQLite schema (events, beacons, tokens, samples, evidence).
- » `dashboard.html` — a dark security-console UI showing live attacker events, beacons, samples, and evidence packages.

Upgrade: the self-defending decoy

Three upgrades extend the honeypot into a self-defending grid, all still collect-and-observe only:

- » `auto_clone.py` — when a decoy draws attention, the grid can spin up fresh, slightly-varied decoys automatically, so an attacker never runs out of "vulnerable" targets to reveal themselves against.
- » `quarantine.py` — the "glovebox." Anything an attacker drops is sealed into an isolated holding area and is never executed — it is only ever read as inert bytes for analysis.
- » `fragment_catalog.py` — catalogs the fragments criminals leave behind (tool signatures, reused credentials, file artifacts) into a growing, searchable record that strengthens attribution over time.

Forensics Lab & Malware Microscope

The forensics lab is the analysis engine. Given a captured sample, it computes cryptographic hashes (MD5/SHA-1/SHA-256), guesses the file type from its magic bytes, measures size, extracts printable strings, and produces a synthetic verdict — Malicious, Suspicious, Low-Risk, or Benign — with a threat score and a family guess. Results accumulate into a searchable threat database (the "virus zoo").

The hard safety line

The microscope never executes the sample. It opens the file and reads its bytes; that is all. Any attribution hint it produces is explicitly labelled **INDICATIVE ONLY** — NOT proof, because attackers routinely route through innocent third parties. The lab's job is to give investigators a well-organized starting point, not to name a culprit.

Data poisoning & integrity

Because the platform builds a growing database of what it analyzes, the integrity of that database matters. Every sample record and every evidence package is content-hashed, so any later tampering is immediately detectable. This is the platform's answer to "data poisoning" — the AI-security risk where an attacker quietly corrupts the data a system relies on. SENTINEL does not defend a single vertical's data; it protects the integrity of its own evidence chain by design.

Air-Gapped Encrypted Vault & the Tier Model

The vault stores sensitive files, notes, and credentials fully encrypted and offline. It uses AES-256-GCM for authenticated encryption, with key derivation scaled by tier (PBKDF2 for lighter deployments, scrypt for stronger ones). Every item is referenced by an opaque token, so even listing the vault reveals nothing about its contents. The prototype's self-test suite passes 23 of 23 checks.

The three tiers, technically

TIER	WHO	KEY DERIVATION	AIR-GAP POSTURE
Family	One person or household	PBKDF2, single passphrase	Software air-gap; local-only binding
Business	SMB / organization	scrypt, per-user keys	Dedicated offline host; controlled sync
Government	Agency / critical infra	scrypt, hardware-key ready	Physical air-gap; nuclear-grade isolation

The same code runs at all three tiers — the difference is configuration and physical deployment, not a rewrite. That is what lets SENTINEL follow a customer from a family plan up to a government contract without changing products.

Filter & Track-Back Gateway

The filter gateway is the protective layer that sits in front of the user — the digital equivalent of a virtual card number that absorbs fraud before it touches your real account. Suspicious links and attachments are "detonated" in a simulated sandbox and dangerous links are swapped for safe, tokenized ones before anything reaches a real inbox, phone, or business file.

Components

- » Filter service (port 5001) — a POST /scan endpoint that inspects a link or attachment in a simulated sandbox and returns a verdict plus a sanitized, tokenized replacement link.
- » Phishing-simulation trainer (port 5002) — a consent-gated training tool that teaches staff to spot social-engineering attempts. It uses no real payloads and cannot be pointed at anyone who has not opted in.

This is where the "phishing scan" and "malware detection" ideas live in the product. Rather than a standalone app, they are the everyday-protection face of the same evidence pipeline: anything the filter catches can be handed to the forensics microscope and, if a real crime occurred, packaged for law enforcement.

Predictive Threat Modeling

A honeypot that only reacts is always one move behind. SENTINEL's predictive layer turns the accumulated history — attacker events, beacon callbacks, cataloged fragments, and sample verdicts — into a forward-looking model of what an intruder is likely to do next, so the deception grid can be positioned in advance.

How it works, honestly

This is pattern analysis over the platform's own logged data, not fortune-telling. By modeling the typical progression of an intrusion — reconnaissance, initial probe, credential attempt, exfiltration of bait, beacon-home — the system can pre-stage the right decoys and raise the right alerts at each stage. The chain below is the sequence the predictive layer scores against:

1

Reconnaissance

Scanners and probes touch the perimeter. The grid logs them and predicts which decoys they will try next.

2

Initial probe

The intruder engages a specific decoy. The system scores intent and can auto-clone additional decoys to keep them engaged.

3

Credential attempt

Login attempts against the fake admin panel are captured. Reused credentials feed the fragment catalog.

4

Exfiltration of bait

The attacker takes a watermarked bait file, believing it is real. The token is now armed.

5

Beacon home

The bait reports back when opened on the attacker's own machine — the lawful track-back moment, now anticipated rather than merely observed.

The Chain of Crime Scenes

A single cyberattack is not one event — it is a series of distinct crimes across distinct locations. SENTINEL treats each as its own preserved "crime scene," then links them into one tamper-evident chain of custody. This is the platform's answer to the founder's vision of following a criminal from the break-in all the way to the resale of stolen goods — lawfully, by documenting rather than pursuing.

CRIME SCENE	WHAT HAPPENED	EVIDENCE SENTINEL PRESERVES
1 · The breach	Unauthorized access to a decoy system	Full interaction log: IP, time, tools, credentials tried
2 · The theft	Exfiltration of a watermarked bait file	Which token left, when, and the beacon it later fired
3 · The callback	Bait opened on the attacker's own machine	Beacon record: time, reporting network, token id
4 · The money	Ransom demand / payment on a public ledger	Traced (not touched) on-chain flow, handed to LE

The on-chain ransom-tracing module

Crime scene four deserves special care because it is where the founder's original "follow the money" idea could have crossed the legal line — and doesn't. SENTINEL never injects a token into an attacker's wallet, never claws back funds, and never touches the criminal's money. Instead, the ransom-tracing module reads the public blockchain ledger — exactly the technique used by professional blockchain-analytics firms — to follow the flow of funds from the ransom address outward, then hands that trail to the cyber-forensics lab and law enforcement.

The prototype models this end to end on a synthetic ledger: it seeds a fake ransom scenario, runs a taint tracer across the transactions, and generates a chain-of-custody report to an outbox/ folder — all clearly watermarked as demo data, under a prominent "trace, not touch" banner.

The Sovereign Packet Layer

The Sovereign Packet Layer connects SENTINEL to the founder's Sovereign Control Tower vision: a way to treat data packets as jurisdiction-aware, tokenized objects so that an operator can enforce whether traffic is allowed to leave the country, must be inspected, or must be blocked outright. It is honest about what it is — you cannot re-plumb the physical internet, so this is a sovereign overlay network that rides on top of it.

What the prototype actually does

The packet gateway classifies each flow as DOMESTIC or INTERNATIONAL, mints a cryptographic token for it, and applies a policy: ALLOW, INSPECT, or BLOCK. In testing it correctly kept domestic critical-infrastructure traffic on the sovereign overlay, allowed allied exchanges, inspected sensitive international transfers, and blocked attempts to export grid-control schematics or to move data to and from sanctioned destinations.

EXAMPLE FLOW	CLASS	POLICY
Power-grid telemetry → defense logistics (US→US)	Domestic	ALLOW · on overlay
Shared threat-intel feed → allied SOC (US→CA)	International	ALLOW
Encrypted research dataset → partner (US→DE)	International	INSPECT
Grid-control schematics → sanctioned dest.	International	BLOCK

This layer is where SENTINEL (defend and track-back) meets Sovereign Control Tower (govern where data may go). Together they form the national cyber-lockdown vision documented separately — the packet layer decides what may leave, and SENTINEL catches and documents whoever tries to break the rules.

Security Model, Legal Guardrails & Deployment

Non-negotiable guardrails (enforced in code)

- » No hack-back. Nothing in the platform reaches into, alters, or damages a system SENTINEL does not own — consistent with the CFAA (18 U.S.C. §1030).¹
- » No live malware execution. Captured samples are read as inert bytes only; the quarantine glovebox never runs anything.
- » Synthetic data only. Every demo uses clearly-labelled fake data, documentation-range IPs, and demo passphrases.
- » Local-first. Services bind to 127.0.0.1; there is no cloud dependency or telemetry.
- » Trace, not touch. On-chain analysis reads the public ledger; it never moves or seizes funds.

Honeypots, beacons, and canary tokens are explicitly recognized as lawful active-defense tactics precisely because they operate only on the defender's own infrastructure.² SENTINEL is built to stay entirely on that side of the line.

Running the prototypes

```
honeypot/ → pip install flask; python demo_seed.py; python honeypot_server.py
(http://127.0.0.1:8080/dashboard)
vault/ → pip install -r requirements.txt; python demo_seed.py; python app.py
gateway/ → filter service on port 5001; trainer/ on port 5002
ransom_trace/ → python demo_seed.py (then run the tracer + report)
packet_layer/ → python demo_seed.py; python packet_gateway.py
```

1. Congressional Research Service, "Cybercrime and the Law: Primer on the CFAA." congress.gov/crs-product/R47557

2. Security Today, "Hacking Back: Revenge is Sweet, But is it Legal?" — honeypots & beacons are lawful. securitytoday.com