REACTIVE PROTOCOL INTELLIGENCE

EREMOS

Eremos: Autonomous Swarm Agents for On-Chain Signal Detection

A Reactive Protocol Intelligence Framework

Built for developers who need early signals, before the noise begins.

v.10 - June 2025

Introduction

Eremos is a lightweight, extensible framework for building autonomous detection agents that monitor blockchain environments with high granularity. It is designed for developers who require actionable insights surfaced through deterministic and pattern-sensitive logic, not noise.

Each agent operates independently, executing scoped logic to trace wallet funding paths, detect contract irregularities, and surface meaningful anomalies across the chain, instantaneously in real-time. Outputs are minimal, structured, and purpose-built for integration into custom toolchains or downstream alerting layers.

Eremos is not a dashboard. It is a living framework.

Quiet until necessary.

Mythos & Narrative

Meet Theron - Agent-000
Inspired by swarm logic, machine instinct, and mythic clarity, Theron is the first deployed agent in the swarm.

Theron does not act. He observes.

Pattern-sensitive, modular, and extendable, Theron is a passive agent calibrated for inference and early signal discovery. It does not operate on volume, but on shape.

Theron is the prototype, an archetype agent that defines the structural skeleton of swarm intelligence. From his blueprint, new agents inherit logic primitives, shared memory utilities, and signal emission standards.

The mythos of Eremos is intentional. In a space driven by speed, hype, and distraction, we deploy watchers who wait, record, and detect with purpose.

System Architecture Overview

Framework Overview

Eremos is composed of atomic agents, each executing scoped logic and emitting structured data scoped to its intended behavior class.

- **Agent Kernel:** Typed logic containers for event hooks, memory, and scoring functions.
- **Signal Bus:** Lightweight publisher module for formatting, hashing, and outputting signals.
- **Utilities Layer:** Shared interfaces for logging, transaction tracing, wallet graphing, and timing.
- **Mempool & RPC Watchers:** Solana-native event listeners, optionally scoped to funding flows, token deployments, and contract interactions.

Eremos uses dependency injection for component layering, enabling each agent to operate in full isolation or via shared state maps.

Signal formatting is type-safe and schema-aligned. Each signal conforms to a uniform output spec, containing origin agent, confidence, glyph, timestamp, and meta tags.

Agent Mechanics

Agent Lifecycle

Each Eremos agent follows a predictable life cycle:

1. **Initialization** - Load detection parameters and signal interface.
2. **Passive Monitoring** - RPC or mempool polling begins.
3. **Trigger Match** - On-chain activity matches detection criteria.
4. **Scoring Phase** - Behavioral heuristics and confidence models evaluate the match.
5. **Signal Emission** - If scoring threshold is passed, signal is emitted.

Signal Emission Format


```
{
  "agent": "Observer",
  "type": "launch_detected",
  "glyph": "Δ",
  "hash": "sig_c7f9a3d2bc",
  "timestamp": "2025-06-12T04:41:25",
  "confidence": 0.91,
  "metadata": {
  "source_wallet": "6Yxk...P2M8",
  "token_tx": "5gW...pump",
  "bundled_activity": 5,
  "elapsed": "13s"
  }
}
```


Signals can be streamed, logged locally, or pushed to external webhooks.

Signal Intelligence

Confidence Scoring Engine

Signals are not binary events. Each one is scored against a dynamic set of weighted heuristics. The default model includes:

- **CEX-Origin Detection:** Traces known centralized exchange outflows (e.g. Kraken, Coinbase).
- **Latency Models:** Measures delta from funding to contract deployment.
- **Wallet Bundling:** Detects clustering behavior via shared origin, proxy txs, and tagging heuristics.
- **Token Metadata Checks:** Detects common anomalies in token name, supply structure, and pre-mint activity.

Agents assign confidence on a normalized float (0 to 1). This allows external systems to filter signals probabilistically.

Glyph System

Each agent uses a unique glyph ID as a symbolic shorthand. Glyphs are included in the signal for indexing, UI mapping, or trigger logic.

Use Cases

Launch Wallet Detection: In the moments before a token is publicly announced, Eremos can identify the true deploy wallet by tracing CEX funding events, analyzing sniping behavior, and scoring token creation latency. This enables early filtering of fraudulent clones.

Wallet Cluster Monitoring: Eremos tracks entity linkage through behavior modeling, shared tx patterns, and proxy wallet tagging. Agents can score wallet groups based on observed co-activity, including coordination in launches or bridging events.

Anomaly Detection: Custom agents can be designed to flag:

- Repeated failed contract creations
- Rapid interaction loops
- Mint bot activity across multiple chains
- Gas spiking patterns linked to timing exploits

Eremos is not opinionated about usage. It provides you the signals. What you build with them is up to you.

Developer Integration

Getting Started

git clone https://github.com/EremosCore/Eremos.git
cd Eremos
npm install
cp .env.example .env.local
npm run dev

Directory Structure

- **/agents** - Base templates, scoped detection modules
- **/utils** - Shared functions for scoring, logging, and tracing
- **/types** - Global TypeScript interfaces and definitions
- **/scripts** - CLI tooling for signal testing and runtime control
- **/docs** - Reference materials, agent design specs, and signal taxonomy

Developer Notes

- Agents are written in TypeScript with strong typing enforced
- No external dependencies required beyond Solana RPC providers
- Compatible with custom inference layers or alerting systems via webhooks

Roadmap

Phase 1: Core Deployment

- Agent-000 live (Theron)
- Modular agent support
- Signal emission interface finalized
- GitHub and Docs are open for contributors

Phase 2: Visualization Layer

- Real-time swarm viewer (UI)
- Signal registry and glyph index
- Logged memory view for agents in active state

Phase 3: Expansion

- Multi-chain support (EVM and L2 focus)
- Event sourcing architecture
- Public signal stream + API endpoints

Phase 4: Open Agent Registry

- Permissionless agent uploads
- Contributor incentive model
- Signal reputation layer

Final Notes

Eremos is not a product. It is not a protocol in the traditional sense.
It is a system. A living one.

Every detection strengthens the swarm. Every contributor shapes the behavior of the whole. Every signal, emitted in silence, reveals something few were watching for.

Enter the system.
Deploy your own watcher.
Observe, quietly.
React, instantly.

https://github.com/EremosCore/Eremos
https://eremos.io