

Aegis -- execution control for AI agents

Aegis is an inline security sidecar for AI-agent tool calls. The central primitive is an Intent Warrant: trusted identity + declared tool intent + policy constraints become a narrowly scoped authorization checked at the execution boundary. Unsafe tool JSON returns HTTP 403 before it is forwarded. On a supported Linux Aegis Node, a syscall outside the signed warrant is EPERM.

Agent -> Intent -> Warrant -> Runtime/Kernel -> Decision -> Evidence

What Aegis is

- Aegis is an execution-control layer for AI agents, not a generic security dashboard.
- The Rust data plane inspects request and response tool-call JSON before the agent can execute it. Denied userspace calls return HTTP 403.
- Three product surfaces in ARCHITECTURE.md: Aegis Enforce (inline proxy, warrants, userspace and Linux kernel enforcement), Aegis Validate (adversarial catalogs and reproducible benchmarks), and Aegis Evidence (signed decisions, warrant records, auditor exports).
- The sidecar is not an identity provider, endpoint protection product, classifier, firewall replacement, or a guarantee that an agent is safe.

How it works

Rust data plane

Pingora reverse proxy on :8080. Intercepts OpenAI-style, Anthropic tool_use, Gemini functionCall, and MCP tool-call JSON on the request and the model response. Streaming bodies are buffered, then inspected. Live LLM tokens do not go through Python.

Python control plane

FastAPI on :8000 plus Command Deck APIs for policy, Crucible, intel packs, HITL, licensing, and audit export. gRPC PolicyService on :50051. The control plane is not on the LLM hot path.

Command Deck

Operator UI at http://localhost:3000 after start. Protect an existing process, fire a canary, and copy signed decision rows. First boot is audit-only until a paid or trial AEGIS-1 key is pasted.

Userspace layers

Allowlist, verified-role RBAC, sequence, intent (including lexical ..), signature pack, multi-turn taint, HITL, and flow/taint. Signature matching is normalized substring + token Jaccard on a pack, not an embedding model.

Optional kernel boundary

On Ubuntu 24.04 Aegis Node, an Ed25519 Intent Warrant binds declared tool intent to later syscalls; mismatch is EPERM. Docker Desktop and Windows stay userspace 403s. A userspace 403 is not kernel success.

Attach without rewriting the agent

Windows: attach.bat support-bot -- python your_agent.py macOS/Linux: ./attach.sh support-bot -- python your_agent.py

The wrapper preserves the agent's vendor key and injects base URLs plus a stable session identity. Raw HTTP clients and programs that bypass the configured proxy are not protected by userspace inspection. Verify routing before making a security claim.

Deployment options

Docker Compose self-hosted

Same public zip on every plan. Windows: AEGIS Suite.bat. macOS/Linux: ./start.sh. Or docker compose up --build from the repository. The base stack runs userspace protection with eBPF disabled, no host Docker socket, no added data-plane capabilities, and no writable kernel mounts. UI and proxy ports bind to localhost.

Aegis Node appliance

Copy the same unzipped folder onto Ubuntu 24.04 (Multipass, a cloud image, or an existing VM -- not Docker Desktop WSL). ./start-node.sh or ./appliance/multipass-launch.sh. The appliance overlay is privileged with host PID and kernel mounts; run it on a dedicated Node VM, not a shared host.

AWS Marketplace

The live SaaS listing is pay-as-you-go. A separate BYOL path is a customer-owned Ubuntu 24.04 Aegis Node: the buyer launches the image in their VPC and pastes an AEGIS-1 key. Cloud-commit buyers use a marketplace private offer (floor \$25k) -- that is not self-serve Stripe checkout. Stripe Builder / Standard / Enterprise stay on Checkout.

Plans and current list prices

Three plans. Same zip. 14-day trial on every tier, card on file. Downloads boot Free Audit Mode (WOULD_BLOCK while still forwarding). Paid or trial keys turn on live 403s and, on Aegis Node, kernel EPERM. The paid key sets the licensed node cap (1 / 5 / 25).

Builder	\$30/mo	\$300/yr	1 node
Standard	\$99/mo	\$990/yr	5 nodes
Enterprise	\$499/mo	\$4,990/yr	25 nodes
Builder	14-day trial, card on file; Live HTTP 403 on this node; Kernel EPERM on Ubuntu 24.04 (not Docker Desktop); Command Deck + attach wrapper; Ed25519 decision records		
Standard	Everything in Builder; 5 licensed nodes; RBAC + sequence + intent; Crucible canary forge; Flow / taint (PCI, PII)		
Enterprise	Everything in Standard; 25 licensed nodes; Air-gapped signed intel + CRL; Guided 30-day POC via sales; Business-hours support (support@aegiscruc.io)		

Optional 48-hour AI agent containment audit: \$4,500 flat, credited toward an Enterprise license if you sign within 30 days. Laptop 403s stay free to prove yourself.

Support summary

support@aegiscruc.io - Monday-Friday, 09:00-18:00 US Central. This is not a 24x7 SOC. Stripe Checkout should use this mailbox and the product name Aegis -- not a silent personal account.

P1	Production enforce path down, or signed keys cannot activate on a paid node	4 hours during support hours
P2	Enforce works; a control is degraded (HITL, intel pack, Command Deck)	1 business day
P3	Install, attach, or docs questions	2 business days
P4	Feature request	5 business days

Do not email the raw AEGIS-1 key. Use the recover URL. We do not promise FedRAMP, FIPS, multi-AZ failover, or kernel enforcement on Docker Desktop WSL.

What Aegis does not claim

- Aegis currently makes no claim of SOC 2, ISO 27001, ISO 42001, FedRAMP, HIPAA, or FIPS 140 certification. The repository is not an ATO package.
- Deployment alone is not a claim of compliance. Control mappings and live-fire evaluations are evidence, not a certification.
- Single-node Compose is not HA. We do not promise multi-AZ failover.
- Kernel enforcement is Linux-specific (Ubuntu 24.04 Aegis Node). Docker Desktop / WSL remain userspace-only.
- Prompt-only jailbreaks that never invoke a tool are signatures or nothing. Aegis is not a model-refusal leaderboard.
- ARCHITECTURE.md states a sub-50 ms target on the userspace signature path. The landing-page figure "0.71 ms" is not treated as a reproduced benchmark in the working paper. Measure on your host.
- evals/enforcement_benchmark.py is a deterministic reference workload with explicit SIMULATED provenance. Do not present simulated or inferred values as live enforcement evidence. Never combine SIMULATED, INFERRED, and LIVE results into a single security percentage.
- Free Audit Mode can log WOULD_BLOCK and still forward. That is not fail-closed.
- This is not a 24x7 SOC, not a third-party pentest of your estate, and not a claim that your agents are safe.