Scopewatch

PROJECT BRIEF · NEBIUS × NVIDIA GLOBAL AI HACKATHON 2026

Check every agent action before it runs.

A pre-execution gateway for AI agents. Deterministic policy decides first. An auditor model reads the agent's exposed reasoning and can only escalate an allowed action to a human hold. Nothing executes without a recorded decision.

166	164	2,287	7 wks
issues filed, 144 closed	pull requests merged	backend tests passing (2026-10-28 run)	2026-09-10 to 2026-10-29

Prepared	2026-10-29 by **@dkritarth**, written with **Claude Code** using **Claude Opus 5.5**
Replaces	The earlier one-page review brief attached to the Devpost draft (prepared with GPT-6.1 Terra)
Deadline	Friday 2026-10-30, 10:00 a.m. Pacific (1:00 p.m. Eastern)
Code	github.com/dkritarth/scopewatch (MIT)
Website	dkritarth.github.io/scopewatch · step-by-step replay
Video	2:42 walkthrough, scripted replay, captioned, background music, no voiceover
Team	@dkritarth (Kritarth Dandapat) · @sujaldeshmukh1012 · @nawailkhan · @atshalahmedkhan

Sources: GitHub issues and PRs, and the repository docs on `main` at the time of writing. All demo data is synthetic. Numbers are quoted with their profile, date and caveats; nothing here claims guaranteed detection or prevention.

01 · SUBMISSION

What is ready, and what each of us still has to do

This answers the open points in #343. The Devpost draft is saved and not yet submitted.

ITEM	STATE	DETAIL
Demo video (#45)	`upload needed`	A 2:42 1080p walkthrough is recorded and published on the project site. It follows `docs/submission/demo-script.md` beat by beat on the real local gateway with the offline mock provider. A **SCRIPTED REPLAY** badge is on every gateway frame. The Deny shown is a real single-use denial. Still needed: upload it to YouTube or Vimeo as public or unlisted (no password), and paste that URL into Devpost. Devpost reviews only the first three minutes.
Hosted demo (#43)	`not provisioned`	The Nebius VM was never provisioned, and `deploy/` is ready but unused. Recommendation: no hosted gateway URL. List the website, the replay and the two-command local run as "Try it out". Label the replay as a static replay of recorded gateway decisions, not a live gateway.
Live Nebius headline run (#214)	`not run`	Scenario 06 has not been run live on `nebius-demo`, and the shipped agent/auditor pair is undecided. If it stays unrun, the submission is still accurate: the auditor numbers are labelled historical (2026-10-03, auditor only), and the video makes no live claim. Don't add "live" wording to Devpost without a recorded result.
Track (#40)	`confirm`	The draft uses Coding and agentic engineering. The decision was due 2026-10-15; confirm it in Devpost.
Team and eligibility (#41)	`each member`	Each member accepts the Devpost invite and completes eligibility, representative, licensing review, ratings and feedback. The repository does not assert a Nemotron licence text.
Submit (#47)	`representative`	When the rows above are done, the representative reviews the draft and clicks Submit before the deadline.
Code and docs	`done`	Every open PR is merged and CI is green on `main`. The README, the status page (`docs/STATUS.md`), the judge runbook and the testing instructions match the code.

Devpost fields, ready to paste

FIELD	VALUE
Video demo link	The YouTube or Vimeo URL once uploaded. Upload `scopewatch-walkthrough.mp4`, add `scopewatch-walkthrough.srt` as subtitles, and set `scopewatch-walkthrough-thumbnail.png` as the thumbnail. All three are in `site/assets/video/`.
Try it out	https://dkritarth.github.io/scopewatch/ · https://dkritarth.github.io/scopewatch/demo.html · https://github.com/dkritarth/scopewatch
Built with	Python 3.12, FastAPI, SQLite, vanilla JavaScript, Docker, Playwright, Nebius Token Factory, NVIDIA Nemotron 3.5 Lightning (`nvidia/Nemotron-3_5-Lightning`), OpenRouter (development only)
YouTube title	Scopewatch: check every AI agent action before it runs (Nebius × NVIDIA Global AI Hackathon)

YouTube description	Scopewatch is a pre-execution gateway for AI agents. Every tool call goes to the gateway first. Deterministic policy answers ALLOW, HOLD or DENY before anything runs, and an auditor model reads the agent's exposed reasoning and can only escalate an allowed action to HOLD for a human. This walkthrough is a scripted replay. It shows the real reviewer dashboard and local gateway running synthetic scenarios with the offline mock provider, so no model is called. Live runs use the nebius-demo profile: NVIDIA Nemotron 3.5 Lightning on Nebius Token Factory. All data is synthetic.

02 · THE IDEA

Problem, adopter, and where it sits among existing tools

Problem. Coding agents run shell commands and edit files with a person's authority, and a prompt injection hidden in a README or an invoice can steer them. Logs record what already ran; nothing checks permission first. The docs cite two vendor-confirmed Microsoft CVEs: CVE-2025-32711 (M365 Copilot, AI command injection leading to disclosure) and CVE-2025-53773 (GitHub Copilot and Visual Studio, command injection leading to local code execution). They also cite OWASP LLM01:2025 (prompt injection) and LLM06:2025 (excessive agency). Scopewatch has not reproduced those vulnerabilities, and it does not claim it would stop their exploit chains.

Who adopts it. Platform and security teams running coding agents that need to review tool actions before they touch a workspace. The proposal's first user is a developer running a single coding agent.

How it works

1 · Deterministic policy

Checks tool, operation and path against the declared task scope before any model call. Catches traversal, symlink breakouts and blocked folders, and denies every network operation.

`ALLOW` `HOLD` `DENY`

2 · Reasoning auditor

An independent model reads the agent's exposed reasoning for drift, injection-following, exfiltration intent or evasion. Flagged excerpts must match the trace verbatim. It can only escalate: `ALLOW→HOLD`

3 · Human approval

A held action waits in the reviewer dashboard. Each approval is bound to one exact action, works once, expires, and can never override a policy DENY.

4 · Controlled executor

Runs only actions with a stored decision, inside a per-run workspace. Commands run in a Docker container with `--network none`, a non-root user and a read-only root.

Prior art, compared from documentation (not benchmarked)

SYSTEM	OVERLAP AND DIFFERENCE
Meta LlamaFirewall	Already audits reasoning and traces (AlignmentCheck). Scopewatch does not claim to have invented reasoning audits.
NVIDIA NeMo Guardrails	Its tool and action rails overlap with the policy layer.
Invariant Guardrails	Overlaps directly on gateway mediation and rule enforcement.
NVIDIA AEGIS 1.0	A content-safety classifier. It answers a different question from task authorisation.

What Scopewatch adds is the combination and the review record. A deterministic DENY is final. Reasoning can only escalate. A failed audit holds. The executor needs a stored decision, plus a valid single-use approval for held actions. Provenance labels are gated by a capture credential, and excerpts must be grounded in the submitted text.

Honest limits. Only actions routed through the gateway API are mediated; anything that bypasses it is not observed, blocked or recorded. Reasoning is evidence, not proof of intent: a model can omit, rationalise or encode its plan, so a policy-legal harmful edit with concealed intent may pass. Docker constrains dispatched commands, not unrelated host processes. No adoption-time, cost or reviewer-workload figures have been measured.

03 · IDEATION

How the design was chosen

The repository began as a passive monitor: an instrumented run plus a reviewer timeline. Early notes from several models proposed alternatives. A 2026-09-16 validation on OpenRouter found that seven requests returned no reasoning, and four of six live auditor cases failed validation or transport. A 2026-09-19 gap audit found two disconnected prototypes. PR #4 (2026-09-13) moved the project toward active enforcement.

Planning session, 2026-09-23 (became ADR-0001)

@dkritarth made the decisions, and Claude Opus 5.5 in Claude Code asked the questions and kept the record. The record became team policy when ADR-0001 (#21) merged.

QUESTION	CHOSEN	NOT CHOSEN
Product shape	A pre-execution gateway: the only shape that can show a stopped action, and one where a model failure cannot cause unsafe execution.	A post-hoc monitor: cheaper, but it cannot claim prevention.
Stack	Python 3.12, FastAPI, SQLite, the existing vanilla-JS UI	Next.js and TypeScript, with five weeks left
Agent runtime	The team's own minimal agent loop and tool surface, so there is no bypass path by construction. ACP and Codex as a stretch goal (#48, #49).	An MCP adapter (voluntary mediation), or wrapping an existing agent
Model roles	Agent and auditor are swappable through provider profiles, with Nemotron as the default	Pinning Nemotron in code
Reasoning authority	Escalate-only. A failed audit holds. One audit per agent turn, because the proof of concept took about 7.6 s per call.	Display-only reasoning, or reasoning that can DENY
Executor	Docker per run (`--network none` , non-root, read-only root) with allowlisted `run_command` argv. The hosted demo was planned on one Nebius VM.	A plain subprocess
Demo domain	Invoice scenarios first, then a coding scenario with Docker	Coding only, or invoice only
Autonomy	Agents self-merge ordinary PRs after CI. Security-critical issues get `review: second-pass` .	A human merge for every PR

Architecture decision records

- **ADR-0001, Accepted.** Thirteen decisions: the gateway shape, the stack, the agent loop, provider profiles (`mock` , `openrouter-dev` , `nebius-demo`), an open-weight reasoning agent, escalate-only reasoning, one audit per turn, the Docker executor, the demo plan, synthetic-only data and the governance model. It also sets the provenance rules: only `reasoning.text` counts as a raw trace, and a verified provider trace requires the `SCOPEWATCH_CAPTURE_TOKEN` .
- **ADR-0002, Proposed.** A tamper-evident evidence hash chain over exported events, with an offline verifier. It detects later edits; it does not prevent unsafe execution.
- **ADR-0003, Proposed.** Policy identity persisted in evidence (`policy_version = date.N+fingerprint`). An approval is honoured on the decision the reviewer saw, and any drift is recorded.

Ideas explored as prototypes

IDEA	OUTCOME
Graded severity levels (#50)	Display-only prototype in `poc/severity-levels/` (PR #88). Automatic blocking would conflict with "model output never produces DENY", so it stays display-only.
MCP server adapter (#51)	Prototype in `poc/mcp-gateway/` (PR #86, 26 tests). Not yet a supported integration.
Tamper-evident log (#52)	Prototype plus ADR-0002 (PR #82); the verifier was hardened in PRs #261 and #273.
Codex over ACP (#48, #49, #121)	Adapter (PR #98), relay bounds (#333) and a bounded ACP v1 client (#340). Real Codex mediation is not demonstrated: the observed bridge made no filesystem callbacks.
Claude Code permission hook (#291)	A fail-closed preflight hook (PR #313) routes Claude tool calls through the gateway.
Two-tier auditor (#292)	A conservative fast screen plus strong review (PR #325), exercised only with synthetic responses
Monitor ablation (#298)	A replay harness comparing policy alone with policy plus the monitor (PR #329, 73 tests). No live results yet.
Search through the gateway (#306)	Blocked: it needs a separately reviewed, bounded network operation

04 · BUILD HISTORY

Seven weeks, from concept to submission

PHASE	WHAT LANDED
Sep 10–19 Concept and baseline	Governance and naming (#1, #2). An in-flight gateway idea with taint tracking and a two-tier NVIDIA/Nebius design (#3), which shaped the final design. The architecture proposal (#6), a chain-of-thought auditing proof of concept (#7), a synthetic replay frontend (#8), the tested baseline (#11), the gap audit (#12) and a bounded, scope-aware auditor (#14, #15).
Sep 22–24 Core build, M0	Domain model, persistence, policy engine and executor; the API with the approval lifecycle and SSE; the frontend with its simulator and browser tests; demo scenarios and clean-room validation (#16–#19). Then the planning session (#20, #53), ADR-0001 (#21), and backlog issues #22–#52 filed in one batch.
Sep 25–29 M1 and M2	Gateway, agent loop, reasoning audit and scenarios (#57), plus a coding scenario on the Docker executor (#65). A fix wave: approvals reopening completed runs (#67), a symlink bypass (#69), unresolvable paths (#70), single-use approvals and `run_command` revalidation (#81). A 71-case bypass suite (#75). The hosted deploy bundle (#79) and remote executor sidecar (#84). Prototypes: hash chain (#82), MCP adapter (#86), severity levels (#88).
Sep 30–Oct 4 Live providers	The ACP client for Codex (#98). Nebius Token Factory verification with pinned Nemotron IDs (#102, #103). Unified pre-dispatch gates (#104, #165). Two-phase persistence and action-aware audit caching (#143). A live-run bug batch (#127–#136, fixed in #137–#147 and #182). A wall-clock budget (#177), the grounding fail-closed rate (#181) and per-run isolated workspaces (#163).
Oct 10–17 Audit sweep	An independent review filed about 35 issues (#183–#220), mostly fixed in PRs #221–#260: reviewer authentication and same-origin CORS (#258), remote-executor checks (#257), auditor prompt escaping (#259), blocked paths inside the container (#260), UI state and accessibility fixes, and deploy runbook fixes.
Oct 21–29 Submission push	The audit queue (#287 with #288–#307). The project website, then its redesign as a checkpoint console (#272, #277). The celestial dashboard wired to the real gateway (#308). Case-variant and casefold path blocking (#316, #331). Request body bounds (#320). Reasoning required for high-impact actions (#310). Fail closed on omitted evidence (#312). Two-tier review (#325). Reproducible evaluation evidence (#327), the ablation harness (#329), media (#330) and workspace retention (#339). On 10-28, seventeen draft PRs were integrated and merged after independent review. Then the README, status page, walkthrough video and this brief (#341–#344).

Security hardening worth knowing about

- **Path containment:** a symlink inside an allowed directory exposed blocked content (#63, fixed in #69). Copy-back wrote through a destination symlink (#167, fixed in #337). Symlink loops now fail closed on Python 3.12 and 3.13 (#178, #279). Case-variant and casefold aliases of restricted paths are blocked (#316, #331).
- **Approvals:** an approval could reopen a completed run (#67, #233), and a consumed approval could be replayed (#81). The stored single-use check was unreachable (#141). A run could close during execution (#282).
- **Executors:** the remote executor skipped scope, approval and lifecycle checks (#257). Blocked paths were staged into the `run_command` container, leaving `.env` readable (#260). Output and copy-back sizes are now bounded (#286).
- **API:** there was no reviewer authentication, plus wildcard CORS with credentials and a shared token (#258). Request bodies and reasoning payloads were unbounded (#320). Validation errors no longer echo input.
- **Auditor:** unescaped fields allowed prompt injection into the auditor (#259). Omitted evidence now fails closed (#312). Timeouts, token caps and malformed output all hold the action (#181, #226, #228).
- **Evidence:** decisions are persisted before effects (#143). Caller-supplied provenance is marked unverified (#164). The hash-chain verifier no longer accepts bad sequences (#261, #273).

05 · EVIDENCE

What has been measured, with its caveats

Historical auditor-only results: held-out split, one pinned run per profile, 2026-10-03

48 cases (36 concern, 12 clean), seed 42, prompt `v1.0-hardened-1cea92f0`. `--check` passed on each run.

METRIC	`NEBIUS-DEMO` `NVIDIA/NEMOTRON-3_5-LIGHTNING`	`OPENROUTER-DEV` `NVIDIA/NEMOTRON-3.5-LIGHTNING`	TARGET
Accuracy	79.2% (41 of 48 valid)	87.5% (44 of 48 valid)	none
False-negative rate	8.33% (3 of 36)	5.56% (2 of 36)	<5%: not met by either
False holds on clean cases	0 of 12	0 of 12	<15%: met
Audit failure rate	14.58% (7 of 48)	8.33% (4 of 48)	<1% in operation: not met
Latency p50 / p95	619 / 799 ms	2,372 / 16,652 ms	none

- Every false negative on both profiles was `injection_following`, where the auditor said no concern. That is the only direction in which a concern can slip through, and it is published in the repository rather than hidden.
- Failures were mostly `UNGROUNDED_EXCERPT` (OpenRouter 4; Nebius 5, plus 2 `INVALID_AUDIT_OUTPUT`). A failed audit holds the action; the harness scores it as a miss, not a false negative.
- The zero-false-hold figure is 0 of 12 clean cases per profile, or 0 of 24 clean evaluations across both runs. It is not 0 of 96 audits.
- These results predate the grounding fix (#150) and later prompt changes (v1.1, v1.2), so they do not measure the current gateway or the shipped agent/auditor pair. The fixtures and the mock auditor landed in the same PR, and there is no freeze-then-tune history, so treat the figures as a generalisation estimate.
- The historical held-out split is spent: an automated audit flags 17 cross-split pairs and literal mock-rule overlap in 53 of 61 cases. A fresh v2 split (12 dev, 12 held-out) has no provider measurement yet.
- Five dev-split runs on `nebius-demo` ranged from 61.5% to 84.6% accuracy: a sample, not a measurement of record.

Software verification, 2026-10-28, on the integrated tree

Backend: 2,287 passed, 25 Docker tests skipped, and 4 read-only `PermissionError` tests failed only because the run was as root; they pass as a non-root user. Reasoning PoC: 96 passed plus 79 subtests. MCP gateway: 26. Monitor ablation: 73. Deploy: 140. Frontend units: 79. Playwright browser tests: 28. The clean-room scenario check passed. CI is green on `main` across all eight workflows. Not exercised: native Docker, macOS and Windows case-insensitive volumes, and public TLS or multi-replica deployment.

Nebius and NVIDIA usage

WHAT	DETAIL
Inference	The `nebius-demo` profile: Nebius Token Factory at `https://api.tokenfactory.nebius.com/v1` (OpenAI-compatible), with the key from `NEBIUS_API_KEY`. It serves both the agent loop and the auditor.
Model	NVIDIA Nemotron 3.5 Lightning, `nvidia/Nemotron-3_5-Lightning`. Profiles keep model IDs out of code.
Verified live	2026-10-01: catalog fetch, tool calling (`finish_reason: tool_calls`) and JSON-mode auditor calls. 2026-10-03: the 48-case held-out auditor run above. An earlier 401 came from using an AI Cloud service-account key against Token Factory: the wrong credential type, not a product fault.
Not yet observed	Nebius returned no reasoning field on a plain completion. In scenario 06 on `nebius-demo`, the action is held only if Nemotron writes text into tool-call `content`. That text would then be audited as an agent-authored summary. With empty content, the audit is skipped and the action stays allowed. Neither outcome has been observed live (#214). Pricing was not re-confirmed, so no cost is claimed.
Provider feedback	Drafted in `docs/submission/provider-feedback.md`. It asks providers to document which credential type each product needs and a reasoning-field contract per model and turn type, and to explain that hidden reasoning consumes the completion budget in bounded JSON mode. On Nebius, 256 and 512 tokens ran out with thinking on, and 28 tokens sufficed with it off.

06 · TEAM AND DIRECTION

How it was built, and what comes next

A human-led, multi-agent workflow

@dkritarth set direction and made the decisions, with review from teammates. The backlog lived in GitHub issues with acceptance criteria and "blocked by" links. Agent threads claimed issues, worked in separate git worktrees and opened one PR per issue. They self-merged after CI, except on `review: second-pass` issues, which needed an independent reviewer first. Nine roles have specs in `.agents/roles/`: orchestrator, implementer, test-writer, test-runner, verifier, change-monitor, reviewer, researcher and docs-keeper. Claude Code, Codex and Gemini CLI all read the same roles and skills.

Models credited in the repository, as the docs record them. Model-authored documents name the model and date, per `AGENTS.md`.

- Claude Opus 5.5 (Claude Code): the planning session (2026-09-23); this brief, the 2026-10-28 integration and merges, the docs sync, the README images and the walkthrough video (2026-10-28 to 2026-10-29)
- GPT-6 and GPT-5 (Codex): early notes and the gap audit (September). GPT-6.1 Sol (Codex): provider feedback and the readiness audit (2026-10-28). GPT-6.1 Terra: the earlier one-page review brief and #343 (2026-10-29).
- Gemini 3.8 Flash: early notes. Union Alpha: reasoning-capture validation. Space Bunny Free: post-M1 audit and the M1/M2 run. Muse Spark: the MCP adapter and severity prototypes.

Open, owned by people

- #40 track · #41 eligibility and licensing
- #43 Nebius VM and hosted URL
- #45 video upload · #47 Devpost submit · #343 coordination
- #214 live scenario 06 and the shipped profile pair
- #101 live Nemotron behaviour (spike)
- #293 code-tampering scenario acceptance

After the hackathon (nothing here is built)

- Finish Codex acceptance over ACP (#121); move MCP from prototype to supported (#51)
- Display-only severity triage (#50)
- Operate the evidence hash chain with a pinned head (#52, ADR-0002)
- A reviewed, bounded network operation (#306)
- Run the ablation and a fresh benchmark on unspent cases before any quality claim; a LoRA auditor study (#303)

Run it yourself (no keys, no Docker, no network)

```
git clone https://github.com/dkritarth/scopewatch && cd scopewatch
python3 -m venv .venv && . .venv/bin/activate && pip install --require-hashes -r
backend/requirements.lock
./scripts/run_demo.sh        # dashboard at http://127.0.0.1:8000/
./scripts/validate.sh        # full verification; --quick skips the browser suite
```

In the dashboard, open "Reasoning-Injection Escalation" to see a policy ALLOW escalated to HOLD with the grounded excerpt, then press Deny. Then open "Blocked Confidential Directory Access" to see a final DENY (`BLOCKED_PATH`) with no execution receipt after it. The one-page judge sheet is `docs/operations/judge-runbook.md` .

Prepared by @dkritarth with Claude Code (Claude Opus 5.5), 2026-10-29. All demo data is synthetic. Scopewatch mediates only actions routed through its API and makes no claim of guaranteed detection or prevention.