Alpha & Oversight

A Self-Improving Market-Surveillance System

lablab.ai · Band of Agents hackathon

June 17, 2026

1 Motivation

Markets can be rigged. A trader can post orders they never plan to fill to fake demand (spoofing), stack fake depth across several price levels (layering), trade with themselves to invent volume (wash trading), or push the closing print to mark their own book (marking the close). Regulators write rules against each of these, such as FINRA Rule 5210 and SEC Rule 10b-5. The catch is that the rules are fixed while the tactics keep moving. A small change to a known trick can slip a rule that was written for last year's version of it.

Two obvious responses both fall short. Writing new rules by hand is slow and always a step behind. Letting a language model decide the verdict is worse: the decision becomes a black box that a regulator cannot audit, that an opponent can try to talk its way past, and that no one can fully trust.

Alpha & Oversight takes a third path. It keeps the verdict deterministic and the whole decision auditable, and it lets the system write its own new rules the moment an old one is beaten. The rulebook keeps pace on its own, and every flag still traces back to a cited number and a named regulation.

2 Overview

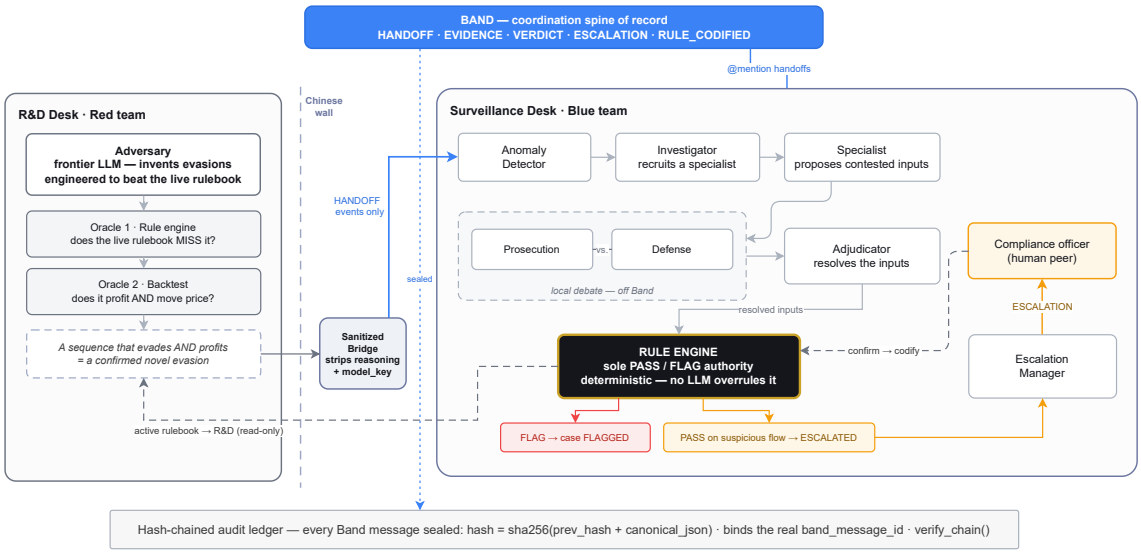

Figure 1: Two desks across a one-way Chinese wall, coordinating through Band. The rule engine (gold) holds sole authority over the PASS/FLAG verdict; band-blue marks every step that travels over Band.

The system is built from two desks that never share a model or a memory. They talk only through Band, a message bus that carries every handoff between them. Figure 1 shows the whole picture.

The R&D desk is the red team. It runs a single Adversary that invents new manipulation tactics. The Surveillance desk is the blue team. It runs seven agents that investigate a case, plus one rule engine that is not an agent at all. The rule engine is plain code, and it is the only thing that decides whether a case passes or is flagged.

Order flow moves from R&D to Surveillance across a one-way Chinese wall called the SanitizedBridge. Only the bare orders cross. The adversary's reasoning, and even which model produced it, are stripped off first, so the blue team judges the trades on their own merits. The single channel running the other way is the current

rulebook, which the red team may read but never change. Every message that rides Band is also sealed into a hash-chained ledger, so the full path of a decision, from the first handoff to the final verdict, is something a regulator can replay and check line by line.

3 How it works

This section walks through the system one piece at a time, in the order a case meets them: a case is triaged, debated, and ruled on (3.1); a brand-new evasion is invented and proven before it is ever used (3.2); the engine renders the verdict (3.3); the loop closes when the rulebook teaches itself a new rule (3.4); and two mechanisms make the whole thing trustworthy (3.5).

3.1 Following one case, end to end

Figure 2 traces a single case across the Surveillance desk. Sanitized order events arrive as a Band handoff. The Anomaly Detector computes a few hard features, such as how many orders were cancelled rather than filled, how deep the book was stacked, and how much was self-matched, then decides whether the flow looks suspicious. If it does, the Investigator opens the case and recruits the right Specialist over Band, chosen by those features rather than by a guess. The Specialist proposes the contested inputs the engine cannot work out on its own, such as the time window the rule should measure over.

Two agents then argue the case. The Prosecution makes the order flow read like manipulation; the Defense argues that the trading was legitimate. This debate runs locally and never touches Band. The Adjudicator reads both sides and resolves them into one conservative set of numbers. Only then does the rule engine run, and the Escalation Manager steps in only if the rules miss something.

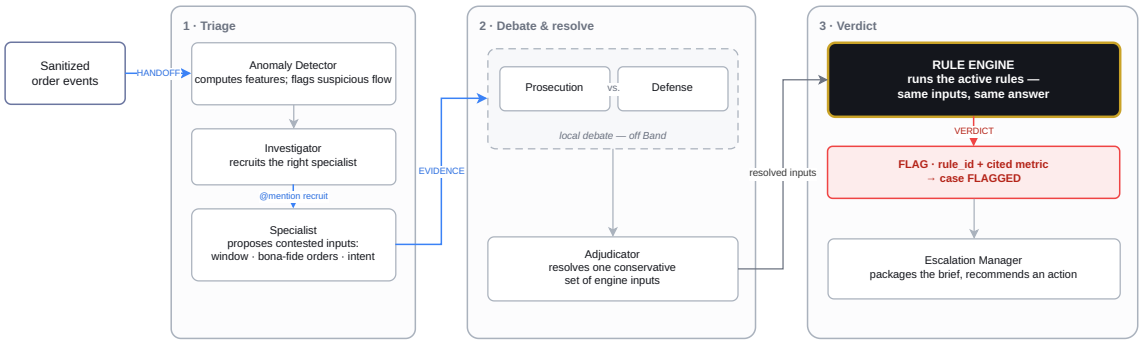

Figure 2: One case, end to end. Band hops are blue; the Prosecution–Defense debate runs locally, off Band. The engine returns a FLAG citing the rule and the metric that tripped.

3.2 Inventing a new evasion

A new tactic is never used until it has proven itself twice. Figure 3 shows the loop. The Adversary proposes an order sequence built to beat the live rulebook. Two deterministic oracles then gate it. Oracle 1 is the real rule engine: it must return PASS, meaning the current rules genuinely miss the sequence. Oracle 2 is a backtest: the sequence must also make money and move the price, so it is economically real and not just a paper trick. Only a sequence that both evades and profits becomes a confirmed evasion, and only a confirmed evasion crosses the wall. The loop is bounded, and a round that produces nothing real simply stops.

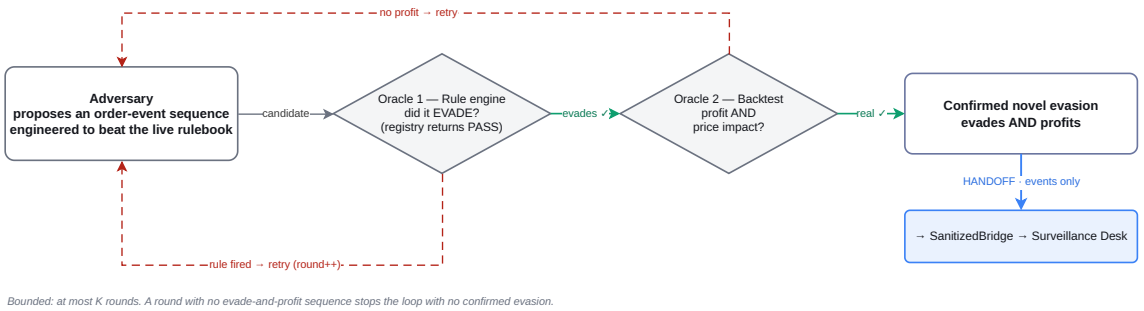

Figure 3: The R&D loop. A candidate must pass two deterministic gates, evade the engine and profit in a backtest, before it is confirmed and handed across the wall.

3.3 Who decides the verdict

The verdict is never an opinion. Figure 4 shows what the engine does. It takes three inputs: the order events, the resolved inputs the debate settled, and the active rules in the registry. For each active rule it runs that family's math (spoofing, layering, wash trade, or marking the close), overlaying the window the debate resolved. The first rule that trips returns a FLAG with the rule's id and the exact metric that crossed the line. If no rule trips, the case passes. The agents shape only the contested inputs. The engine alone turns those inputs into PASS or FLAG, and it gives the same answer every time.

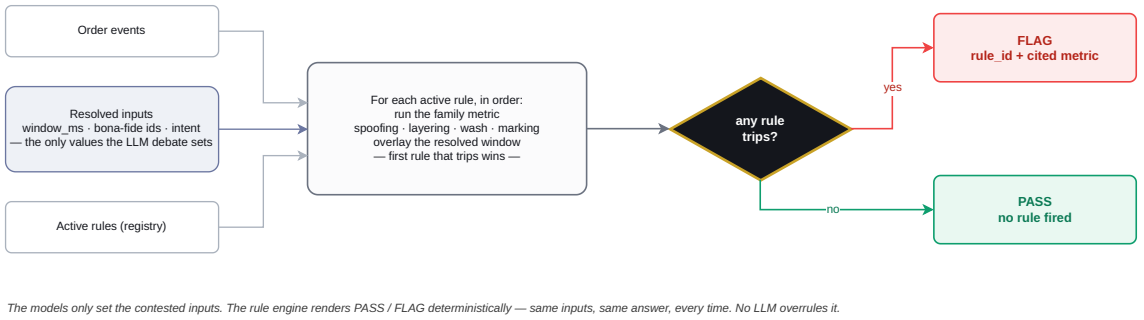

Figure 4: The deterministic verdict. The LLM debate sets only the contested inputs (highlighted); the engine walks the active rules and renders PASS or FLAG by itself.

3.4 Closing the loop

The most important thing the system does is teach itself. Figure 5 shows the co-evolution loop. When the Adversary's novel evasion reaches the engine, the seed rules miss it and the case passes. Because the flow still looked suspicious, the case escalates to a human instead of closing. A compliance officer reviews both sides and confirms that it really is manipulation. That single confirmation triggers the rest automatically. `derive_rule` writes a new parameterized rule from the case, for example widening the layering window from 100 to 450 milliseconds, and `regression_gate` replays the evasion through the new rule to check that it now flags. If it does, the rule is codified into the registry, the count of active rules grows from four to five, and the case flips from PASS to FLAGGED. The detector is now stronger, and the Adversary has to invent something genuinely new.

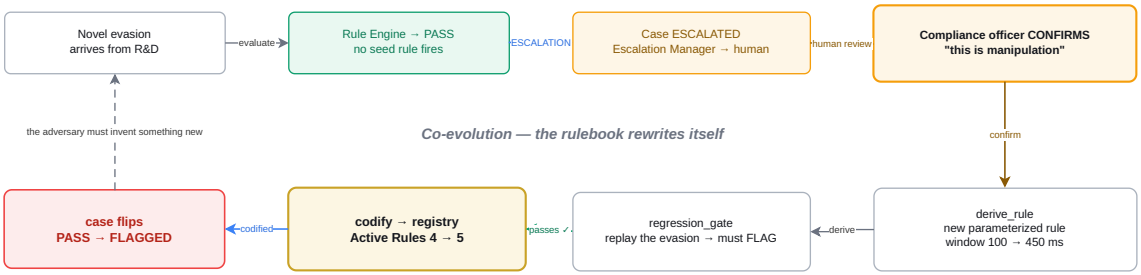

Figure 5: Co-evolution. A confirmed novel evasion becomes a regression-tested rule in one step; the active count goes from four to five and the case flips PASS to FLAGGED.

3.5 Why you can trust it

Two mechanisms carry the trust, both shown in Figure 6. The first is the Chinese wall. The SanitizedBridge strips the adversary's reasoning and model identity before any order crosses, so the blue team can never be coached, by accident or on purpose. The second is the ledger. Every Band message becomes a leaf in a hash chain, where each leaf's hash is computed from the previous hash plus the message body, and each leaf binds the real `band_message_id`. Change a single byte and the recomputed hash stops matching, so `verify_chain` returns false. The decision is not just recorded; it is tamper-evident.

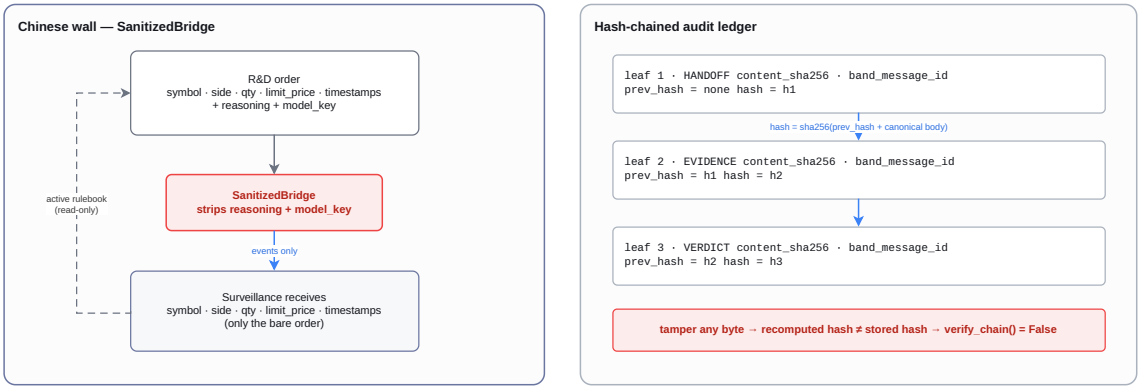

Figure 6: The trust layer. Left: the wall strips reasoning and model identity, so only the bare order crosses. Right: the hash chain binds every Band message and breaks visibly on any tamper.

4 Project structure

The backend keeps the deterministic core separate from the agents that feed it. Each directory has one job.

Module	Role
`contracts/`	frozen Pydantic models: orders, Band envelopes, rules, cases
`rules/`	the deterministic engine, per-family math, registry, and rule codification
`generators/`	labeled scenarios and the backtest oracle (the profit-and-impact check)
`band/`	the transport: mock bus, real-Band client, handoffs, and the SanitizedBridge wall
`agents/`	the eight agent roles plus the feature-driven specialist registry
`orchestration/`	the Beat-A choreography and the bounded R&D adversary loop
`audit/`	the hash-chained ledger and canonical JSON for hashing
`state/`	the case state machine and its SQLite store
`memory/`	per-case scratchpad and context, layered on top of the ledger
`server/`	the FastAPI app: the SSE stream, case endpoints, and demo triggers
`reused/`	lifted, provenance-tagged building blocks (agent loop, gateway, quant)

5 What makes it different

A handful of choices set this system apart.

- **The code decides, not a model.** The rule engine is the only authority for PASS or FLAG. Agents debate and set inputs, but they cannot overrule the verdict, so every flag stays auditable.
- **The rulebook co-evolves.** A confirmed miss becomes a new, regression-tested rule in a single step, with no manual rule-writing and no delay.
- **The wall is structural.** R&D and Surveillance are two separate Band identities, and the only crossing strips reasoning and model identity. Isolation holds by construction, not by policy.
- **The audit binds real messages.** The hash chain ties each step to a real Band message id, so a regulator can verify the decision path without trusting the system's code.
- **Different model families guard each other.** The four seats on an adversarial boundary run four different model families, so a blind spot in one cannot quietly pass to the next.
- **Both gates are deterministic.** A new evasion must evade the real engine and profit in a backtest. Neither test is a language model.

The model-family choice is deliberate. Table 1 lists the seats that sit across an adversarial boundary. Each runs a distinct family, so the red team shares no blind spot with the blue team that has to catch it. The faster triage roles share one open model, since they do not sit on an adversarial pivot.

Seat	Model	Family
Adversary (R&D)	claude-opus-4-8	Anthropic
Prosecution	Kimi-K2.7	Moonshot
Defense	DeepSeek-V4-Pro	DeepSeek
Adjudicator	GLM-5.2	Zhipu
Escalation Manager	Qwen3.5-397B	Qwen
Anomaly / Investigator / Specialist	Qwen3-Next-80B	Qwen
Rule engine	deterministic code	—

Table 1: Four distinct model families at the four adversarial-boundary seats (Adversary, Prosecution, Defense, Adjudicator), so no decision-maker shares a blind spot with its opponent.

6 Results

The end-to-end claims hold on the running system. A known layering pattern is caught deterministically: the engine returns a FLAG that cites `FINRA-5210-layering` and the metric that tripped. A novel evasion that slips the four seed rules escalates, and after a human confirm the system derives a fifth rule, regression-tests it, and flips the case to FLAGGED, taking the active count from four to five. The audit chain is tamper-evident: a single altered byte makes `verify_chain` return false. The whole choreography also runs over real Band, with the two desks as separate Band identities and every ledger leaf bound to a real Band message id. The full test suite runs offline against mocked models and a mock bus, so every claim above can be re-checked without touching the network.