Trust No Single Witness

Differential fuzzing & bug-bounty triage for Ethereum clients

Bhargava Shastry · Protocol Security, Ethereum Foundation · Berlin Ethereum Day

𝕏 @ibags · GitHub @bshastry

> There is no ground truth in Ethereum — only witnesses that disagree, and security is the art of trusting none of them alone.

The closest thing to ground truth — an executable **spec** of the consensus-critical *core* (**execution-specs**, **consensus-specs**) — is *partial* and *fallible*. A privileged witness, not an oracle. Still just one.

A bug is a disagreement

- Ethereum bets on **client diversity** for resilience: no monoculture, no single point of failure.
- The hidden consequence: **no production client you can declare "correct."** There *is* an executable **spec** of the consensus-critical core — **execution-specs**, **consensus-specs** — but it's *partial* and *fallible*: a privileged **witness**, not an oracle.
- So correctness is *relative*. You find bugs by making implementations **disagree**.
- Diversity is the **blessing** (it's the only thing that makes this possible) and the **bane** (N attack surfaces; a disagreement on mainnet is a chain split).

The disagreement machine: goevmlab

- **Differential EVM fuzzing**: generate a state test, run it across **~80 client instances** in parallel, compare state-root + execution-trace hashes. Any mismatch = a candidate consensus bug.
- Clients under test: **geth · Nethermind · Besu · Erigon · reth · Nimbus · evmone · EELS**.
- Run **nightly**, fork-targeted (Cancun → Prague → Osaka), with **Telegram/ntfy telemetry**.

CI for consensus — catch the regression the hour the commit lands.

Caught on introduction

A real divergence, summer 2025 — Osaka fork prep, never reached a release:

- **17 Jul, 01:09** — a client ships a new bn254 elliptic-curve backend. Its point decoder *silently reduces* out-of-range coordinates instead of rejecting them.
- **17 Jul, 13:24** — ~12 h later the nightly run pages: on `ecAdd` (`0x6`) with a non-canonical point, **geth rejects** (`0x0`), **the other client accepts** (`0x1`). Different result, different gas — **a state-root split.**
- **19 Jul** — fix merged: validate on decode. The alerts stop.

A chain-splitting disagreement: found 12 hours after the commit, gone in 48.

Why it works: the interface decides reproducibility

- EVM **state tests** ride a standardized interface (`t8n` / statetest). A divergence is a *file* you replay against every client in milliseconds.
- Standardized repro → fast triage → fast fix. The interface is what makes the machine pay off.
- Bug **shapes** it surfaces: **state-root divergence**, **gas-accounting** divergence, **EIP-7702** set-code/delegation, BLAKE2 rounds. (shapes, not payloads)

The turn: a green checkmark is an agreed-upon guess

The machine only *perceives* a bug as a **disagreement in EVM execution**. Three whole classes fall outside that — and the bounty firehose says that's where the bugs actually are:

1. **P2P attacks** — discovery floods, snap-sync loops, gossip & peer-scoring abuse. *Not in the EVM domain at all — invisible by construction.*
2. **Block-validation bugs** — fork-choice splits, an invalid block accepted, a missing-field block imported. *In scope, but block tests have no standardized replay interface — badly under-fuzzed.*
3. **Shared spec misreads** — the canonicality or fork-gate rule *every* client reads the same wrong way. No disagreement → no alarm. Ever.

~half of recent confirmed bounty true-positives are p2p / networking; almost none are pure EVM-execution consensus — the machine's own home turf.

> Your oracle is blind to the most dangerous bug of all: the one nobody disagrees about.

Attacking your own blind spots

- **Oracles that don't need disagreement:** LLM property-based testing — turn EIP prose into *independent* executable properties (`propgen`: P256, BN254, BLS12-381, ecrecover). Now a bug they **all share** still fails. (spec misreads)
- **Force the rare trigger:** topology-driven generation (8 topologies × 7 scenarios) — catches the detectable bug random generation never builds.
- **Open the frontier:** blocktest / EEST corpus fuzzing — a call to standardize the block-test interface. (block validation)
- **Reactive fuzzing:** a high-profile bug reveals a *class* → targeted fuzzer → sweep for siblings. "Nothing severe" **bounds** the class. *Negative results are results.*

But this is one thread — EVM consensus. Total coverage = internal + external.

The second oracle: the world reports disagreements

Proactive fuzzing was **internal** intel. This is **external** — the whole ecosystem reporting disagreements. The job here is **signal-to-noise** for the client team that actually fixes the bug:

- **Intake** — `bounty-tools`: Google-Form CSV → an append-only `store`, dedup, first-pass screen.
- **Reproduce safely** — `bounty-vm`: two-tier isolation for **untrusted** PoCs — a quarantine KVM/QEMU VM (no network), then per-client Docker built from source at a pinned ref, run `--network=none`.
- **Confirm live** — replay against a multi-client devnet; Engine-API injection turns a decode divergence into a real VALID/INVALID split.

We're broad, not the per-client specialist — we know enough of each client to boost the signal.

The AI turn: AI broke the channel — and is how we cope

- **Post-AI, the bounty channel flooded.** Volume exploded; quality variance widened; reports written in *"classic AI-slop"* phrasing. Triage became **the** bottleneck.
- Fight fire with fire: a parallel agent **severity swarm** rates each report against the EF rubric — and now **runs the PoC inside the `bounty-vm` sandbox** to check it **reproduces** and **matches the reporter's claim**.
- Still **not trusted alone** (false positives *and* negatives): PII-scrubbed input, execution **only** in the air-gapped sandbox, and it **must downgrade to `unsure`** rather than guess.
- **AI rates — and reproduces — the flood that AI created; a human breaks the tie.**

Three witnesses, deliberately in conflict

Every finding carries three independent severity signals, surfaced *on purpose*:

WITNESS	SIGNAL
Reporter	`claimed_severity`
AI swarm	`agent_severity` + **reproduces? / matches claim?**
The network	`blast_radius` = live client-share weighting

- Severity isn't *"it crashes"* — it's **amplification × propagation × asymmetry**.
- Case in point: a real, confirmed single-peer DoS — **rejected** as out-of-scope (~1:1 attacker-to-victim cost, no amplification). **Honest downgrading is the product.**

Triage is a vantage point, not a queue

- **Threat intel:** the aggregate of submissions is a sensor network — it shows which **bug shapes are trending** and which **threats are emerging** across the whole client ecosystem.
- **Variant analysis:** one bug in one client → a *hunting lead* in every other. A fix in geth is a question to ask of Nethermind, Besu, reth.
- The **bane** of client diversity (N codebases) becomes an **asset**: the same logical mistake recurs as variants you can go find.

Trust no single witness

The fuzzer. The AI. The reporter. **Every oracle is fallible.**
You **triangulate their disagreements** — and a **human breaks the tie.**

Diversity makes the hunt possible — and a split catastrophic. Total intel = internal + external; our edge is breadth and signal, the client teams hold the depth and the fix. AI scales the witnesses — it never replaces the judge.

Takeaways

1. A bug is a disagreement — so a green checkmark is an agreed-upon guess.

2. Build oracles that catch the bug everyone shares.

3. AI scales the witnesses; it never replaces the judge.

Bhargava Shastry · find me at Berlin Ethereum Day