StereumLabs

AI-Powered Observability for Ethereum Staking

EthCC[9] | Cannes | March 30 – April 2, 2026

David Mühlbacher | RockLogic GmbH

Who We Are

RockLogic GmbH

Austrian infrastructure company

ISO 27001 certified

Lido Curated Set

Active Node Operator

Running production validators

37 Client Combos

6 EL × 6 CL + Erigon/Caplin

Bare metal, 90-day metrics

We don’t just build monitoring tools. We run validators. Every day. On production infrastructure.

The Problem

Getting answers about Ethereum client behavior is painful.

- Open dashboards, navigate dozens of panels
- Write queries, compare time ranges manually
- Hope you're looking at the right metrics
- **At 3am, you need answers. Not dashboards.**

Client dev ships a release → Did it help or hurt? Across how many EL pairings? Compared to what baseline?

The StereumLabs Platform

20+ dashboards covering all 37 Ethereum client combinations

6 Execution Layer

Geth, Nethermind, Besu,
Erigon, Reth, Ethrex

6 Consensus Layer

Lighthouse, Prysm, Teku,
Nimbus, Lodestar, Grandine

1 Standalone

Erigon + Caplin


Isolated bare-metal nodes


Real, reproducible, comparable data


Ethereum Foundation grant

But dashboards have limits. That's where AI comes in.

AI Chatbot

Natural Language Meets Live Data

```
"Compare disk growth between Geth and Erigon over the last 30 days"
```

```
"How did the Prysm update from v7.1.1 to v7.1.2 affect resource usage?"
```

```
"Which consensus client uses the most bandwidth as a supernode?"
```

```
"Show me Lighthouse memory changes after the latest update"
```

Proper analysis back. With numbers. Across all EL pairings. In seconds.

The Secret Sauce: Our Instruction Set

The AI model alone doesn't produce useful results.

Our Instruction Set is the glue between raw data and useful AI output:

- Which metrics matter for which client combination
- Normal ranges per pairing (3 GB RSS for Prysm+Geth = fine, for Prysm+Besu = red flag)
- Client architecture differences (Go vs. Java GC, Rust memory models)
- Query sequencing: which queries to run in which order

Continuously expanding. Built from years of professional node operation. Can't be replicated by just connecting an AI to a data source.

Proof: Prysm v7.1.1 → v7.1.2

One question to the AI chatbot → Full cross-EL impact analysis

Memory (RSS)	Block Processing	Peer Count	CPU
-5.1%	**-25%**	**~71**	**Mixed**
avg. Geth pairing -8.8%	Erigon: 403ms → 90ms	Stable. No regression.	Reth -21%, Besu +28%

This analysis would normally take hours of manual work. Generated from a single question.

docs.stereumlabs.com/blog/prysm-version-7-1-1-and-7-1-2-comparison-resources

AI Alerting

From “Something Is Wrong” to “Here’s Why”

Stage 1: Near-Real-Time Alerts

Examples:

Attestation rate < 95%

Disk usage > 90%

Peer count drop

Unexpected SSH login

Client service restart

and many more…

Stage 2: AI Root-Cause Analysis

Webhook triggers AI on anomaly

Pulls metrics + logs, correlates data

Root-cause analysis + next steps

Delivered in 5–15 seconds

Not just “attestation rate dropped below 95%” but WHY it dropped and WHAT to do about it.

Built-In Baseline

Instant context from our neutral dataset across all 37 client combinations

Every alert answers:

- **Is this happening across the Ethereum network?**
- **Is it specific to this client version?**
- **Or is it unique to your local environment?**

Without a neutral baseline, operators are guessing. With it, they know.

Security Monitoring

Near-real-time AI analysis of security events

SSH Login Checks

Authorized key? Expected source IP?
Expected time window?

Service Restart Analysis

Config reload → verify fee recipient
address hasn't been changed

Config Drift Detection

Unauthorized processes, port openings,
validator key access patterns

Traditional monitoring: "Geth restarted." Our AI: "Fee recipient changed from 0xABC to 0xDEF. Not your deployment pipeline. Severity: critical."

Deployment Models

SaaS

Hosted by us

Subscribe, we handle everything.

ISO 27001 certified environment.

Best for smaller operators.

Pull Model

Alerting-as-a-Service

You expose Prometheus endpoint.

Your data never enters our systems.

Meets baseline at AI layer only.

On-Premise

Full control

Entire stack on your infra.

Your own API keys.

Data never leaves your network.

"Your data never enters our systems. It only meets our baseline at the AI inference layer."

What Makes This Unique

01 Coverage

37 client combinations on dedicated bare metal.
No one else monitors the full matrix.

02 Instruction Set

Years of node operation distilled into the rules
that make AI output useful, not noise.

03 Security Expertise

General IT infra + crypto. Multiple security audits.
Attack vectors baked into alert rules.

It's not the AI model. Anyone can call an API. It's these three things combined.

Current Status

Dashboards (20+)	Live	All 37 client combinations
AI Chatbot	Working PoC	Running against live production data
AI Alerting	Q2 2026	Two-stage architecture, Stage 1 active
Security Monitoring	Q2 2026	SSH, service restarts, config drift
On-premise deployment	Ready	Full stack deployable on customer infra

Fusaka Hardfork Impact

14-day before/after comparison across 36 non-supernode pairings

Network RX	CPU	Memory	Disk Reads
-60%	**+30%**	**-8%**	**-53%**
2.20 → 0.87 MiB/s	4.66% → 6.08%	6.06 → 5.59 GiB	2.58 → 1.21 MiB/s
PeerDAS in action	Sampling overhead	Less blob data in RAM	Fewer full-blob fetches

PeerDAS trades modest CPU for massive bandwidth and I/O savings. For operators paying per-GB egress, this is a significant cost reduction.

Fusaka: Client Highlights

Notable outliers across consensus and execution clients

Consensus Layer

Nimbus	CPU +257%	Most CPU-intensive PeerDAS impl.
Lighthouse	CPU -13%	Only CC to reduce CPU post-fork
Grandine	RAM -17%	Strong memory improvement
Nimbus	RAM -20%	Largest CC memory drop

Execution Layer

Besu	CPU +101%	Gas limit + EOF overhead
Geth	CPU +64%	Significant increase
Nethermind	CPU -26%	Only EC to reduce CPU
Besu	RAM -35%	Largest EC memory drop

Worst pairing post-fork: Nimbus + Reth at 16.78% CPU — highest of any combination across the fleet.

What would you want to ask your infrastructure?

We're looking for:

- Node operators to try the AI chatbot
- Client teams for automated cross-EL impact analysis
- Staking protocols for monitoring standards

StereumLabs

David Mühlbacher | contact@stereumlabs.com