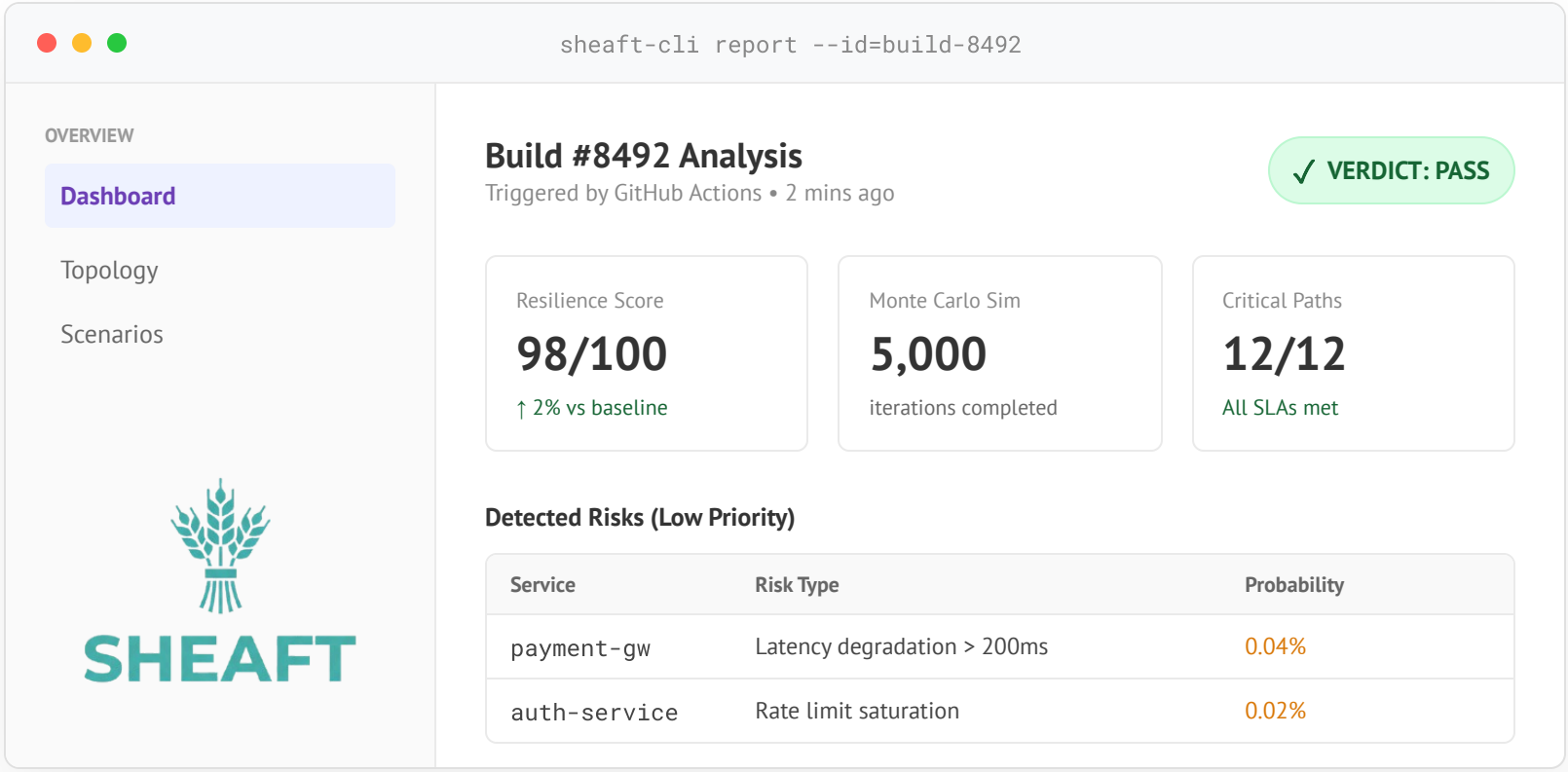
Sheaft

Risk-free virtual crash tests for critical digital infrastructure

Sheaft turns existing telemetry into a resilience digital twin, simulates failures before release, and gives platform/SRE teams a green/yellow/red deployment verdict — without breaking production.



Sample release-risk verdict

Release failures are still discovered in production

Every architecture change can create hidden failure paths, but teams still lack a regular, low-cost, risk-free way to test resilience before release.

What happens today

Reactive fixes

Observability explains what already broke in production. Sheaft answers the question before release: what will break if a dependency fails?

Manual chaos engineering is expensive

Real failure injection requires a safe environment, preparation, and rare expertise, so it is not run on every release.

Manual diagrams become obsolete fast

Microservice dependencies change faster than engineers can update diagrams and risk tables.

Market signals

$300k+
average cost of one hour of downtime for enterprise businesses

57.4%
do not use chaos engineering

23%
write fault-injection scripts themselves

63%
build reliability tools in-house

SIGNAL 1
No market standard

SIGNAL 2
Heavy in-house tooling

SIGNAL 3
Entry window for Sheaft

DevCrowd SRE 2025 survey: overall n=273; chaos engineering question n=183.

Telemetry becomes a resilience digital twin

Sheaft automatically builds a dependency model, runs millions of virtual failure scenarios, and returns a verdict: can critical user journeys survive the next release?

STEP 1

Telemetry

The customer already has traces, OTLP, topology, and configuration data.

STEP 2

Digital twin

Sheaft builds a model of dependencies, replicas, and critical user journeys.

STEP 3

Simulation

Sheaft runs millions of virtual service-failure scenarios in minutes.

STEP 4

Verdict

The team gets a green, yellow, or red release-risk verdict.

What makes it innovative

- Turns passive telemetry into an active resilience test.
- Tests the unreleased change without touching production.
- Replaces rare manual chaos campaigns with a repeatable pre-release check.

Innovation proof points

ICSE 2026 Distinguished Paper Award
international validation of the model-discovery approach

Russian patent application
technology protection in Russia

Registered computer program
software code protection

FSI grant
support for deep-tech development

MTK status
Small Technology Company in Russia

Resilience assessment before release and at runtime

The team sees which operations and dependencies make a release fragile, what should be fixed before deployment, and which user paths became risky in observability mode.

OUTPUT 1

Risk report

A list of critical paths, vulnerable operations, and weak dependencies that can affect the release.

OUTPUT 2

Release decision

A clear next action: deploy, deploy with a warning, or fix the weak points before release.

OUTPUT 3

Observability signal

The calculated resilience state can be returned to the observability stack next to metrics, logs, and traces.

This reduces manual alignment between platform, engineering, and operations teams and helps select live checks more precisely.

Who needs Sheaft inside the company

Sheaft sits at the intersection of release risk, platform delivery, and operational resilience.

Pain owner

SRE / operational reliability / incident management

PAIN

Post-release incidents, hidden dependencies, and unclear blast radius when something fails.

GETS

A list of fragile routes and dependencies.

Adoption owner

Platform Engineering / DevOps Platform / Release Engineering

PAIN

Needs a standard release check without manual diagrams or live chaos for every release.

GETS

A check that can run in shadow mode first and later fit into delivery.

System owner

Backend / product engineering team of a critical service

PAIN

A dependency or topology change can break a user path while normal tests stay green.

GETS

Risk scoring for its operations and clear fix points.

Pilot sponsor

Head of Platform / Head of Engineering / CTO-1 / digital business owner

PAIN

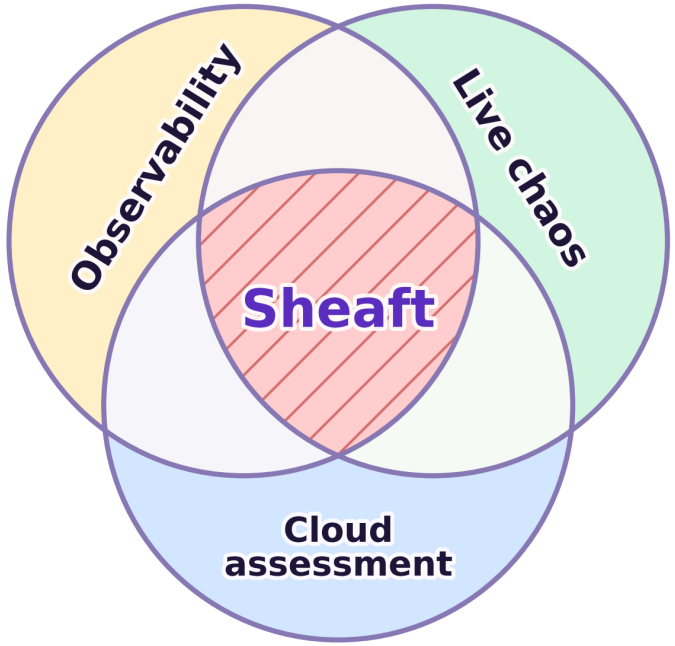
Needs a measurable way to reduce release risk.

GETS

A bounded pilot with clear success criteria.

Ideal first pilot: one critical microservice contour with traces or OTLP, incident history, and a team responsible for releases.

We are forming a new class of solutions



The missing pre-release layer between observability, live chaos, and cloud assessment

Who we *(do not)* compete with

SOLUTION CLASS	WHO BUILDS IT	LIMITATION
Observability (monitoring)	Monq, Proto, Smart Monitor, Yandex Monium, Coroot, Datadog	**Post-incident only.** Shows that the system has already failed in production.
Live Chaos (runtime chaos)	Gremlin, Chaos Mesh, in-house scripts	**Production risk.** Expensive and dangerous to break the real system on every release.
Cloud Assessment (audit)	AWS Resilience Hub, cloud-provider tools	**Vendor-locked.** Static and does not test the code's business logic.
Virtual Chaos	**Sheaft**	**Before release.** **Risk-free.** **Vendor-agnostic.** **Runs on existing telemetry.**

Technology readiness

Public releases, scientific validation, and integration-ready inputs make it practical to verify Sheaft on a real technical contour.

PUBLIC STACK

What is available

Bering v1.0.0 — model discovery with stable model and snapshot contracts, cross-platform binaries, an OCI image, and a Helm chart.

Sheaft v1.2.0 — operation-specific fail-stop tolerance, confidence bounds, and release gates for regressions in the modelled margin.

mb3r-stack v1.1.0 — verified Bering 1.0.0 + Sheaft 1.2.0 bundle with end-to-end checks and an opt-in failure-tolerance profile.

INNOVATION

Scientific basis

ICSE 2026 / NIER Distinguished Paper Award

Model Discovery and Graph Simulation was recognized in the New Ideas and Emerging Results track and demonstrates the approach on DeathStarBench.

AINA 2026 (CORE B)

The OpenTelemetry Demo case clarifies model boundaries and shows the signal on an applied stack.

INTEGRATION READINESS

Ready for existing telemetry

Standard inputs - traces, OTLP, topology, and configuration are enough to build the dependency model.

Reviewable artifacts - model discovery and simulation are split into clear artifacts before any delivery-process automation.

No live fault injection - resilience is evaluated through virtual failures, so production is not deliberately broken.

The pilot starts with historical-data validation

The team provides traces for one contour and incident history for a selected period; Sheaft builds the model and simulations on our side, then compares the signal with real incidents.

Input

What we need from the team

- one service contour where failure or degradation has a visible business cost
- traces, OTLP, or telemetry export for this contour
- incident and degradation history for the selected validation period

Run

What we do

- **without integration** we accept the export and clarify contour boundaries
- build a model of dependencies, replicas, and critical user paths
- run failure simulations and produce a report on weak points

Result

How we decide

- compare detected risks with incident and degradation history
- check which failures Sheaft could have flagged in advance
- choose the next step: recurring run, observability integration, or commercial stage

First step without integration: no agent, CI/CD gate, or production connection is required.

Contacts

Technical walkthrough, project materials, and direct contact with the team for the next implementation conversation.

Anatoly Krasnovsky

Founder / Technology

contact@mb3r-lab.org

+7 923 410-32-25

@ANATOLY_KR

10+ years in backend engineering and architecture. Reliability and graph-modeling expert. Author of the scientific foundation behind the project.

mb3r-lab.org

Documentation, GitHub, and materials for technical assessment and the next conversation.