Savant Chat

AI-native security.

New standards for secure smart contract development.

ALEXANDRA GULAMOVA

savant.chat

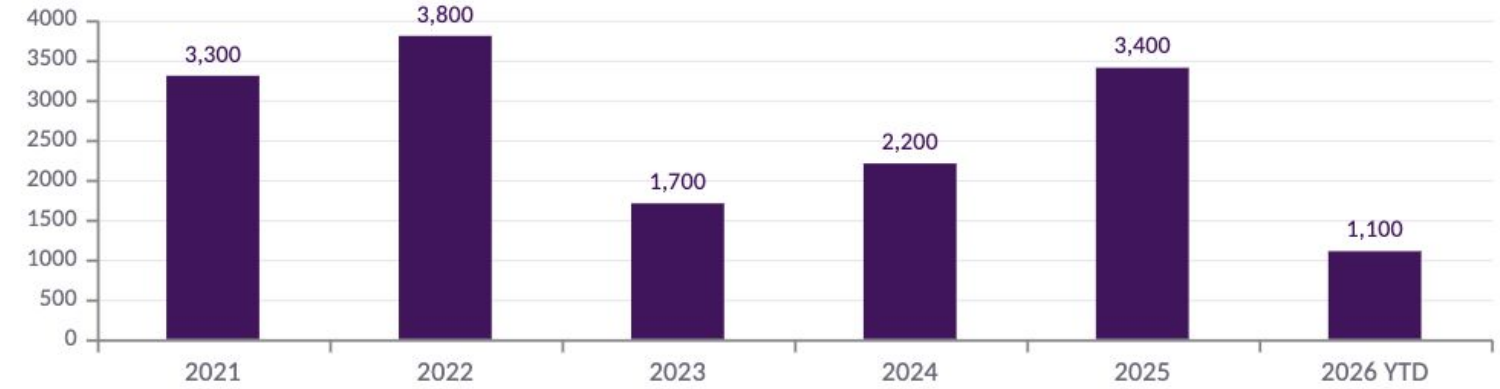
Web3 in 2026 — the crisis hasn't slowed down.

Savant Chat

Annual stolen funds, April 2026 incident log, and a trend nobody is bending.

$3.4B

stolen from crypto in 2025 —
the highest since 2022



April 2026 — a hack almost every other day

Apr 01	Drift Protocol	$285M	6-month social engineering → fake CVT collateral
Apr 10	Rhea Finance	$18.4M	flash-loan oracle manipulation
Apr 14	CoW Swap	$1.2M	DNS hijack / front-end attack
Apr 18	KelpDAO (rsETH)	$293M	LayerZero bridge — single-verifier compromise
Apr 30	Wasabi Protocol	$4.5M	admin key compromise

Yearly totals: Chainalysis Crypto Crime Report 2026. April 2026 incidents: Hacken, Halborn, Chainalysis, MetaMask Security Report.

The trend is up and to the right — but so are the attackers' tools.

Manual audits vs hackers with AI tools.

Savant Chat

An old workflow against an exponentially-faster opponent.

Defenders today

- Read the code line by line
- Bound by human attention span
- Weeks per audit, months in queue
- Cost scales linearly with codebase
- Pattern memory limited to recent work

≈ 1 codebase / week / senior auditor

Attackers with AI

- 10,000+ parallel scans per night
- Pattern-match across every public exploit
- Hours from new pattern → live exploit
- Cost falls as model prices fall
- No NDAs, no review queues

≈ entire chain scanned every 24h

Personalized tools vs universal tools.

Why a brilliant private setup doesn't scale a team.

Savant Chat

Personalized in-house setup

Built by one engineer, for one engineer.

- Works for one head, not for a team
- No bandwidth for proper product research
- Rewritten every time a model ships
- Knowledge dies when the author leaves

Universal AI product

Built by a team whose only job is making it better.

- Hardened on thousands of real codebases
- Full-time research, eval & red-team loops
- Works the same for every team member
- Improvements ship to everyone at once

Whitehats vs the blackhat economy.

Savant Chat

The unit economics are upside-down — and that's the real problem.

At a **0.1% vulnerability rate**, scanning a chain is **profitable for attackers on $6K exploits**. Defenders need **$60K bounties just to break even** on the same scan.

$6K

exploit size at which scanning becomes profitable for attackers

$60K

bounty needed for defenders to break even on one scan

33,000

future scans funded by one $100K successful exploit

3,300

scans a $10K bounty buys a defender (10× less reach)

Defenders can't out-budget attackers — they have to out-automate them.

The evolution of audit automation.

Savant Chat

From human eyes to LLM agents — three generations in a decade.

Each generation didn't replace the last — it stacked on top.

2015 → **2018 →** **2025 →**

Manual review

Eyes on every line. Slow, expensive, but contextual.

Static analyzers

Slither, MythX, Echidna. Fast — but pattern-bound and noisy.

AI tools & agents

LLMs that reason about intent — not just syntax.

What's actually on the table in 2026.

Four flavors of AI in the audit pipeline — and they are not equal.

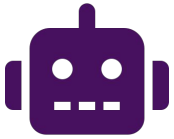
Savant Chat

Stock LLMs

Plain chat with Claude / GPT — copy code, paste, hope.

Claude Skills / Agent kits

Pre-packaged prompts + tool routing on top of a general model.



Specialized AI agents

Purpose-built for vulnerability hunting. Their own scaffolding.

CI / CD integration

Security checks running on every PR — before code ever reaches mainnet.

Stock LLMs.

Cheap to start. Easy to fool yourself into thinking you've configured them.

Paste code → ask for bugs. That's the whole workflow.

Pros

Relatively cheap per token

Pay-as-you-go, no platform fee.

Zero setup — works on day one

Open a chat, paste, go.

Feels customizable

System prompts give the illusion of a tuned tool.

Cons

Slow in practice

Every audit becomes a manual prompt-engineering session.

Highly subjective output

Same prompt, two answers — depends on how you ask.

No memory across the codebase

Each chat starts from zero. No invariants, no context.

Claude Skills & agent kits.

Smarter than raw chat — still spending cognition on the wrong things.

Savant Chat

Pre-packaged prompts, tool routing, file access. A general-purpose model in a costume.

Pros

Deeper than raw LLM chat

Skill scaffolding remembers how to think about a problem.

Lower setup cost per audit

Open the skill, point at code, go.

Composable with your other tools

Slot fuzzers, formal verifiers, internal data sources into the loop.

Cons

Model isn't tuned for bug-hunting

It's a generalist — vulnerability priors are shallow.

Cognition burned on tool choice

Every step it asks itself "which tool now?" instead of "is this safe?"

Hard to evaluate consistently

Output shape changes with prompt drift. Hard to gate on quality.

Specialized AI security tools.

Not every "AI auditor" is one. The wrapper trap.

Tell a wrapper from a real tool.

A real specialized tool can't be cheap. Vulnerability hunting at production scale needs up to

10,000+ parallel queries

to a tier-1 LLM per codebase — plus custom scaffolding, evals, red-team loops and a private benchmark suite. If a vendor charges "a few bucks per audit" — they're reselling the chat window.

How to tell:

- Publishes its own benchmark, not someone else's
- Cost reflects compute, not a flat subscription
- Finds bugs that survived a human audit

Saves the team's most expensive resource — senior auditor hours — for the work only humans can do.

CI/CD integration — security where code is born.

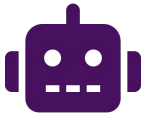
Savant Chat

If a vulnerability ships to mainnet, you've already lost the cheap fight.

01

Pull request opened

Developer pushes a Solidity / Move / Cairo change.



02

AI security gate

Specialized model reviews the diff — context-aware, not pattern-only.

03

Findings inline

Comments on the PR with reproductions, severity, and a suggested fix.

10× cheaper

than catching a bug post-deploy

Hours, not weeks

audit feedback inside the dev loop

Stops regressions

every change is re-checked against the same invariants

Product security in 2026 — the stack.

Three layers. None of them is optional.

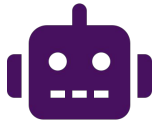
Savant Chat

01

CI/CD security gates

Catch the cheap bugs before they ever leave the developer's branch.

02



AI prep for human audit

Compress the codebase, surface invariants, hand the auditor a head start.

03

Human audit (still essential)

Senior humans on the parts only humans can judge — economic intent, governance, novel attack surfaces.

Fatal mistakes of web3 security in 2026.

Two ways to die confidently.

Savant Chat

Mistake #1 — Treating any one layer as the whole answer.

"We have an audit, we're safe." "We have AI, we're safe." "We have a bug bounty, we're safe." Each one alone is a Swiss cheese slice. Stack them.

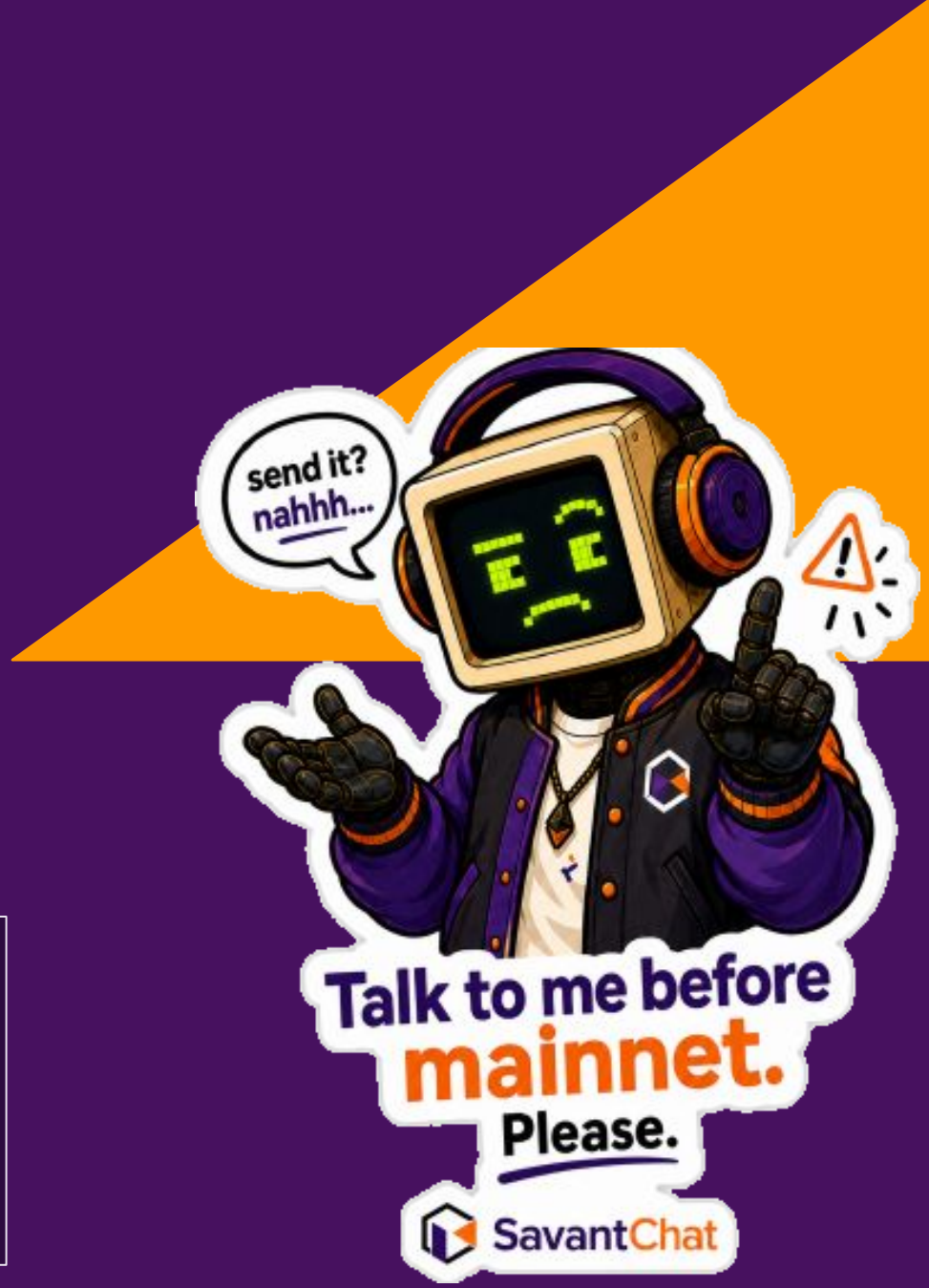
Mistake #2 — Trusting cheap AI tools.

If a tool costs $5 per audit, it ran $5 worth of compute. That buys you a single chat completion — not the 10k-query scan your protocol deserves. Cheap AI is a comfort blanket, not a defense.

Savant Chat

Build like attackers already have AI.

Because they do.



Try Savant.chat

savant.chat
Specialized AI agent for smart-contract audits
82% on EVMbench