Proof of Audits

We find bugs in a contest. Then we keep the proof alive.

Company: Proof of Audits

Document: Company whitepaper

Version: 2.1 · 24 August 2026

Status: Live 30 July 2026. Pre-revenue.

> *Proof of Audits is a Web3 security contest platform. We find bugs with 4x coverage and keep a live page that shows what was checked and if the code still matches.*

This paper describes the product as shipped. It is not a token sale. It is not a safety certificate. The live site is proofofaudits.com.

Abstract

Proof of Audits is a **Web3 security contest platform**.

Teams pay for a contest that finds bugs. We give that contest **4x coverage**: function, contract, connections, and the whole system. Each layer has an owner. Then we keep a **live page** (a Trust Passport) that shows what was checked, what was fixed, if live code still matches, and what is still open.

$16.65B already left on-chain after people trusted a report (DefiLlama). Not because there were no audits. Because a paid review still could not answer a wallet.

You wait, you pay, you get a PDF. The next code change makes that PDF old. Open contests find bugs, then pay copies of the same bug and leave hard parts with no owner. People still sign off a badge.

A firm PDF cannot close this. An open contest cannot close this. 4x coverage, then a live page, is the structure that closes both holes.

We do not claim exploits are impossible. We show proof and gaps.

One sentence: We find bugs in a contest, then keep a live page that still matches the code.

---

1. What this is

Proof of Audits is **first a contest**, then a live page.

Founders already know how to buy a contest: freeze the code, put money in a pool, researchers hunt bugs, payouts settle. We sit on that same shelf as Sherlock and Cantina.

The difference is what the contest is *for*, and what it leaves behind.

Layer	What it is
Contest	4x coverage. Function, contract, connections, whole system. Each layer has an owner.

Proof	A live page: what was checked, what was fixed, if live code still matches, what is still open

We are **not** an audit firm. We are **not** a rating badge. We are **not** a claim that exploits are impossible.

Public definition (also on `/what-is-proof-of-audits`):

> *Proof of Audits is a Web3 security contest platform. We find bugs with 4x coverage and keep a live page that shows what was checked and if the code still matches.*

Who it is for

Actor	Job
Protocol / founder	Buy a contest that maps coverage, then keep a passport investors can open after ship
Auditor	Get assigned scope, base pay for completing it, bounty for unique validated findings, portable score
Investor / LP	See scope, fix state, live match, keys, and gaps before capital moves
User	See a signal before signing, not a logo from last quarter

What people can inspect

Four questions the passport is built to answer:

1. What was reviewed?
2. What was fixed?
3. Does live code still match?
4. What is still open or changed?

If a field is missing, the product is supposed to show the gap, not invent a green badge.

Live product: https://proofofaudits.com · /verify-contract · /deployed/evidence/aave-v4 · /extension

The Aave V4 page is a **public evidence record**, not a customer.

---

2. Problems in contests (the buyer's real pain)

This section is the contest critique. It is why we do not run an open free-for-all and call the PDF done.

2.1 Duplicate pay

In an open contest, a High that 40 people also found is still one bug. Platforms still have to judge 40 write-ups. Prize math splits the same High among the crowd. Protocol money pays for copies. Unique researchers get diluted.

This is not theoretical. Public judging repos routinely list a single High "also found by" dozens of handles. Industry write-ups (for example Zellic's contest FAQ) warn that decks count raw Highs without de-duplication.

What we do instead

- Copied or substantially reproduced findings are rejected, not paid as unique work.

- Auditor reputation uses a **uniqueness multiplier**: solo 1.00, 2-4 dups 0.40, 5-9 0.20, 10+ 0.10.
- Reliability scoring penalizes spray-and-pray (`valid/total`).
- A **3% submission stake** on auditor base discourages junk filings.
- Each selected tier share splits **50% base / 50% bounty**. Base is for completing assigned scope. Bounty is for accepted unique findings. The protocol is not buying 200 copies of one reentrancy.

We still de-duplicate. We just refuse to treat the duplicate pile as coverage.

2.2 No owned coverage

"We ran a contest" does not say whether `deposit()`, the upgrade proxy, or the oracle adapter was actually looked at. Open contests have no function owner. A leaderboard of findings is a list of what people *chose* to write up, not a map of what was reviewed.

Sherlock and peers will say they "handle duplicates for you." That is judging hygiene. It is not a coverage map.

What we do instead

The T4-clustering engine decomposes the repo into review-sized closures (function, contract, cross-contract, system). Bin packing produces `auditor_slot_N.json` packages. **One auditor per cluster.** The record can name the function, the tier, the owner, and the confidence of the parser stack.

This is **assigned review**, not a guarantee that every possible bug was found. Public copy must not say "Coverage Guarantee." The honest claim is: coverage is *mapped and owned*, and missed in-scope bugs can slash the owner.

2.3 Cherry-picking

Open contests pay for findings, not for finishing the boring half of the graph. Rational researchers hit high-expected-value surfaces (vaults, oracles, access control) and skip admin, upgrade, view-heavy, and cross-contract paths that take days and may yield nothing.

That is why a contest can produce "zero valid High/Medium" *and* still leave whole closures untouched. Absence of paid findings is not evidence of review.

What we do instead

- Base fee is earned only if the auditor accepts the slot, reviews the **full allocated scope**, files an original report, and passes closeout checks. No finding in a clean cluster still pays base. Skipping the cluster does not.
- Missed in-scope bugs slash reputation and base (critical –40 rep / 35% base; high –20 / 20%; medium –8 / 10%). Slashed base splits 60% to the finder / 40% to the platform.
- Higher tiers validate lower-tier work. T4 does not get to "call it done" on a function T3 never sees.
- T1 owns system economics, governance, and emergent behavior: the work cherry-pickers skip.

2.4 Other contest failure modes we designed around

Failure	What happens on open contests	What we do
Pool theater	Marketed $2M, unlocked $10k (public complaint in 2025 contest market)	Pool is cluster-priced ($8k-$150k). Base is a real share, not a teaser.
Politics routing	Work goes to brand-name handles	Routing uses verified finding history, tags, tier gates, COI checks

Reputation silos	C4 score ≠ Sherlock score	M3 formula weights platforms equally (1.0). Quality is findings, not brand.
Report dies	Leaderboard post, then silence	Passport + 5-min match poll + authority poll
Judging pile-up	Hundreds of invalid/dup issues	Validation cascade T4→T3→T2→T1, T0 lock, rejected-copy rules

---

3. Problems with proof, for users

Even a well-run contest still fails the person who has to sign or allocate.

A PDF answers "was this reviewed once?" It does not answer the four questions at wallet time (also on `/wiki/the-trust-gap`):

Question	What the PDF usually says	What the user needs
Scope	"Contract X was in scope"	Was **this function** reviewed, at **this commit**, by **this tier**?
Freshness	"Clean at commit abc123"	Does live bytecode still match, including proxy implementation?
Authority	"Multisig" in a screenshot	Who holds admin / pause / upgrade **now**?
Residual risk	"3 findings, resolved"	What was out of scope, unfixed, or unverified after the fix commit?

Loss figures we use as *motivation*, not as a claim that we prevent hacks:

- **$16.65B** cumulative, DefiLlama hacks database
- **$36.7M** from protocol-owned contracts with unverified source (Chainalysis, 2026)
- **$340.7M** across 14 bridge incidents (PeckShield / CoinGabbar, 2026)
- **$1.1B+** 2026 YTD by 11 June (Bitrue)

The pattern is trust extended without inspectable proof at the moment of decision. A badge on a website is not that proof.

Users also hit:

- **Grey silence.** Most contracts have no evidence. A loud false alarm would train people to ignore the tool. The extension stays silent when we have no record.
- **Score confusion.** Mixing a native lifecycle score with an external deployed score is a lie. We keep Native v5 **/1400** and External ITS **/900** on separate scales.
- **Screenshot abuse.** A green number without scope, date, and stale state is the old badge problem. The passport has to show blockers and gaps on the same page.

---

4. Clarity (what "audited" has to mean here)

We replace a slogan with a checklist. If we cannot fill a row, that row is a gap.

Must be clear	Where it lives
Locked commit / scope boundary	Pre-audit
Which functions / contracts / paths	Clustering + Function Review Lens
Who reviewed which cluster	Slot assignment
Finding, PoC, validator, not self-attested	Core audit + T0
Fix commit, replay, independent verdict	Post-audit
Live bytecode vs reviewed baseline	Deployment Match
Admin, pause, upgrade	Authority map
What is still open	Passport gaps / Sentinel
Which score family you are looking at	Native /1400 vs External /900 vs 0-300 deployed

Design rules (enforced in product copy too):

- Evidence over assertion. Nothing important is "trust us."
- Show gaps. Drift is flagged, not hidden.
- Banned phrases: "100% secure," "certified safe," "no risk," "perfect score," "final security guarantee," "Coverage Guarantee."
- Point-in-time honesty. A review is valid for a commit and a moment.
- Formulas are version-tagged (`2026-05-v2`) so old scores can be recomputed.

---

5. The pipeline

Contest-style discovery sits inside five stages. Public wiki: `/wiki/how-it-works`, `/wiki/pipeline`, `/protocols/pre-audit`, `/protocols/core-audit`, `/protocols/post-audit`, `/protocols/scorer`.

```

PRE-AUDIT → CORE CONTEST (T4→T1 + T0) → POST-AUDIT → SCORER → PASSPORT

lock, map, assigned clustered review prove fixes /1400 match + signal

cluster, brief

```

5.1 Pre-audit (prepare)

Lock the source snapshot. No moving target during review. Build the risk map, invariant registry, quality gates, T4 clusters, and `core_audit_brief.md`. Auditors do not start cold.

If gates fail, admin actions that would open core review are blocked.

5.2 Core contest (discover + validate)

Code is clustered. Each unit has a boundary, a risk/effort score, and an eligible owner. That is 4-layer coverage: function, contract, integration, system. Four hunts, not four copies of the same file. Cherry-picking dies when each surface has an owner.

Tier	Public name	Owns
T4	Swordfish Scout	Function-level hunt; local invariants; focused PoCs
T3	Hammerhead	Whole-contract structure; validates T4
T2	Orca Warden	Cross-contract paths, oracles, dependencies; validates T3
T1	Phantom Octopus	System economics, governance, MEV, emergent behavior; validates T2
T0	Turtle Arbiter	Appeals, missed-bug attribution, sanitized report, final lock

T4 work is not done until T3 (or the configured validator) accepts it. Findings flow up. The same reviewer cannot validate their own finding. T0 locks the report. Settlement inputs come from the locked record, not from a chat thread.

5.3 Post-audit (prove the fix)

A fix is not "fixed" because the protocol said so.

Independent panel: diff fingerprint, finding replay on the fix commit, regression / invariant rerun, alternative paths, verdict (accepted / incomplete / regressed / residual). Only then is the commit a candidate for deployment match.

5.4 Native scorer

Approved lifecycle evidence becomes **Native v5 /1400**: buckets, caps, blockers, warnings, maturity. Admin publish gate. This is not External ITS /900 (deployed-protocol assessment) and not the 0-300 per-address score the extension queries.

5.5 Deploy match + Trust Passport (protect after ship)

Compare live bytecode (and EIP-1967 / beacon proxy implementation) to the reviewed baseline. Poll match on a short interval (Sentinel: code ~5 min, authority ~15 min). Publish the passport. Wallet-time overlay can read the same record.

Publication pipeline in the wiki: Audit (T4-T1) → Gate (post-audit) → Publish (passport) → Credential (evidence).

---

6. How the contest is actually scoped (clustering)

Routing needs units of comparable risk and effort. The engine is graph-theoretic, not ML.

```

Repo intake → Parser stack → Validation gate → Closure builder

→ Scoring → Escalation → Bin packer → Line extractor → Scope packager

```

Solc AST is ground truth. Slither, tree-sitter, and coverage are enrichment. Confidence: HIGH ≥95% analyzer coverage, MEDIUM 75-95%, LOW 50-75%, FAILED <50%. Production runs often lean on solc AST; confidence is conservative when enrichment layers are thin. That limitation is public on `/wiki/clustering`.

T4 cluster score: `cluster_score = round(0.65·R + 0.35·E, 2)` with R clamped 150, E clamped 100.

Risk examples: `delegatecall` +85 (×3), `tx.origin` / `selfdestruct` +85 (×3), upgrade pointer write +48 (×2.5), assembly +38, admin write +38, oracle write +32, unguarded external call +22, guarded call –14.

Effort: loc, branches, loops, external calls.

Higher tiers use the same idea with different weights. Packing is First-Fit-Decreasing, 15% overshoot, default threshold 150. Auditor count `ceil(total_score / threshold)`. Each slot carries `estimated_hours = max(total_score/10, 2.0)`.

Protocols select depths: T4 only = 25% of pool, through all four = 100%. They pay for the layers they buy. They do not get to pretend T4-only was a full-system review.

---

7. How auditors are scored (so routing is not politics)

7.1 M3 contest score (`2026-05-v2`)

```

ContestScore = Σ(FindingScore_i) × RankBonus × Difficulty × TimeDecay

FindingScore_i = SeverityBase × Uniqueness × Outcome

```

Piece	Rule
Severity	Critical 15, High 8, Medium 3, Low 0.5, gas/QA/info 0
Uniqueness	solo 1.00 · 2-4 dups 0.40 · 5-9 0.20 · 10+ 0.10 · unknown 0.60
Outcome	paid 1.00 · disputed-upheld 0.90 · accepted zero-payout 0.70 · pending 0.50 · rejected 0.00
Rank	top 1% 1.50 · 5% 1.30 · 10% 1.15 · 25% 1.05
Difficulty	`min(2.50, 1+ln(max(participants,2))/8)` × sparsity (1.30 if <5 unique findings)
Time	6 months 1.00 → 18 months 0.80 → 36 months 0.60 → 60 months 0.45 floor
Platforms	Sherlock = Code4rena = Cantina = Spearbit = Trail of Bits = 1.0
Reliability	valid/total gates after ≥5 submissions

VAPS (public 400-1000): finding quality ~35%, market validation ~20%, repeatability ~20%, recency ~15%, breadth ~10%.

```
VAPS = clamp(400 + min(raw,10000)/10000 × 600, 400, 1000)
```

7.2 Promotion (one-way)

T4→T3: reputation ≥50, tasks ≥10. T3→T2: 150 / 30. T2→T1: 300 / 60.

Guards: ≥5 validated findings or cap at T4. Zero native (in-platform) findings cap at T3. T2/T1 admin-gated; T1 needs ≥2 T1 vouches. Cooldowns 60 days (T2) / 90 days (T1). Sybil hard-cap 70. SERIALIZABLE row locks on tier changes.

Calibration: "certain" + rejected = –4; "possible" + confirmed = +3. Overclaimers are visible.

---

8. Living proof surfaces

8.1 Trust Passport

One live page: scope, auditors by tier, findings, fix state, code match, authority, **open gaps**. Shareable. Not a safety label.

8.2 Function Review Lens

Maps a wallet action (deposit, withdraw, borrow, stake, claim, bridge) to the exact function, reviewing tier, confidence, residual risk. "Audited" is a codebase claim. The lens asks whether **this call** was reviewed.

8.3 Deployment Match

Expected vs observed bytecode hashes per address. Proxy-aware. Drift flagged. This is the $36.7M unverified-source class: users should see a mismatch, not a stale PDF.

8.4 Contract Shield (extension)

Manifest V3. `storage` only. Wraps `eth_sendTransaction`, `eth_signTransaction`, `wallet_sendCalls`. Calls `POST /api/public/extension/analyze` (9s, cache ~5s). Green / yellow / red when we have evidence; **grey / silent** when we do not. Does not persist calldata, addresses, or signatures. EIP-6963 provider overlay.

Supported check surface: Ethereum, Base, Arbitrum, Optimism, Polygon, BNB Chain (as shipped).

8.5 Score families (do not mix)

Score	Range	Use
Native v5	/1400	Native lifecycle evidence for a Proof of Audits contest/engagement
External ITS	/900	Deployed-protocol assessment (separate family)
Deployed protocol score	0-300	Per-address score the extension can query
VAPS	400-1000	Auditor display score

Older docs mentioned a Native VTI /1200 scale. **Public native max is /1400.** Do not rescale one family to look like another.

---

9. Economics (as sold, lock file)

Three-sided: protocols pay, auditors earn, investors and users read proof. **No token is required for the beta product.** Fees are quoted in USD / stablecoin.

From the company money lock (do not invent other numbers in outreach):

Item	Rule
Extension score	Free
First gap map (live teams)	Free

Native contest	$8,000-$150,000 by cluster band
Protocol platform fee	3% of core, **outside** the auditor pool
Auditor service fee	5% on auditor payouts
Live follow-on	Pay only to audit **confirmed holes**, then Native /1400 scorer $3,500 / $5,000 / $6,500
18-month plan	100 protocols, ~$1.35M (plan, not booked revenue)
Market framing	$2.31B now, $6.84B by 2030 (security market, not our revenue)

Cluster bands (illustrative, from clustering): 0-8 clusters ~$8k … 120-200+ ~$100k-$150k.

Per selected tier share (wiki `/wiki/compensation-rules`):

```
selected_tier_share = core_pool / number_of_selected_tiers

base_pool = selected_tier_share × 0.50

bounty_pool = selected_tier_share × 0.50

gross_base_per_auditor = base_pool / assigned_auditors_in_tier
```

Bounty planning split of the tier bounty pool: Critical 35%, High 30%, Medium 25%, Low 10%. Accepted finding rates against task base: Critical 40%, High 22%, Medium 12%, Low 4%, Info 0.8%.

T0 reviewer: 2% of selected tier shares. Submission stake: 3% of auditor base. T0 appeal stake: 5% of base, returned on successful appeal.

Raised: $0. **Cap table:** 100% founder. **Customers / trailing revenue:** none. Do not invent LOIs, CoinGecko/CMC as live partners, or Aave as a customer.

---

10. Architecture (as shipped)

```
Browser / wallet → Next.js 16 frontend + Contract Shield

HTTPS

Fastify + TypeScript API

■ ■

PostgreSQL 16 Redis 7 (BullMQ)

(source of truth)
```

T4-clustering is a Python process (`networkx`, `py-solc-x`, Slither). Child products (Sentinel, stake-to-audit, war room, and others) are **separate processes** on the shared DB + HTTP API. They do not import core internals.

Backend is fail-closed on missing critical tables. Migrations are ordered and idempotent.

---

11. Competition

Alternative	What they sell	Where they stop
Audit firms (Trail of Bits, OpenZeppelin, CertiK)	Deep snapshot PDF	Point-in-time; no function-level living match for users
Contest platforms (Sherlock, Cantina)	Parallel bug discovery	Report / leaderboard; duplicates and cherry-pick are structural; output ages on the next commit
Monitoring / ratings (Forta, Skynet, Defender)	Alerts and scores	Do not prove which function was reviewed at which commit
Explorers (Etherscan, Sourcify)	Source ↔ bytecode	No audit-scope provenance, no residual-risk page
Status quo	Badge, PDF, or skip	Blind signature

We compete for the **contest budget** first. The proof layer is why a team should pick this contest instead of (or after) a pure open contest. Firms can still do a private deep pass; we complement that with assigned contest density plus living proof.

Incumbents can copy a passport UI. Harder to copy: assigned clustering, uniqueness-weighted reputation across platforms, and a decision-time record that compounds per version.

---

12. Why now (company-safe)

Protocols ship continuously. Audits and contests remain point-in-time. Upgradable proxies and post-review diffs mean teams, LPs, and users need to know **which functions were reviewed, whether deployed bytecode still matches, and who holds authority.**

That is the living proof layer. Timing is operational, not a slogan about "audited contracts still get hacked."

Traction we are allowed to state: product live 30 July 2026, 40+ auditors scored, founder outbound. Not revenue. Not logos.

Next 90 days (lock): first paid gap audit, 10 founding protocol baselines, keep checker and extension live. Path signal: a protocol pays to close the free gap map, then returns on the next version.

---

13. Limitations

- Not a safety guarantee. Evidence, gaps, risk signals.
- Assigned coverage is not complete discovery. Missed-bug slash reduces, does not eliminate, the cherry-pick problem if validators fail.
- Match proves a code relationship, not that configuration, keys, or integrations are safe.
- Parser enrichment (Slither, forge coverage) can be thin; confidence stays conservative.

- Reputation ingestion trusts source platforms, then applies our formula. Garbage upstream stays garbage unless rejected.
- Child products (bounty, war games, academy) are mixed maturity. This paper is about contest + proof.
- Scores are bounded heuristics with version tags, not oracles.

---

14. Summary

Open Web3 contests find bugs. They also pay duplicates, leave coverage unowned, and let researchers skip the unsexy graph. The PDF they produce cannot answer a wallet or an LP at decision time.

Proof of Audits is that contest category, rebuilt around **owned clusters**, **base pay for finishing the slot**, **uniqueness-weighted bounty**, **slash for missed in-scope work**, and a **passport that still matches live code**.

If we do the job, a founder can say more than "we were audited." They can point to who reviewed which function, whether the fix commit closed the path, whether mainnet still matches, and what is still open.

See the proof. Know the risk. Then decide.

---

15. Public documentation map

Google Search Console export `proofofaudits.com-Coverage-Valid-2026-08-24` listed **91 valid** indexed URLs (chart date 21 August 2026). Use these as the HTML twin of this paper. Do not invent extra live partners from this list.

Whitepaper topic	Live page
What it is	`/what-is-proof-of-audits`
This paper (HTML / PDF / markdown)	`/whitepaper`, `/whitepaper/download`, `/whitepaper.md`
Contest vs PDF	`/audit-pdf-vs-trust-passport`
Trust gap	`/wiki/the-trust-gap`
Pipeline	`/wiki/how-it-works`, `/wiki/pipeline`
Clustering	`/wiki/clustering`, `/wiki/closure-building`, `/wiki/bin-packing`
Pay / dups / slash	`/wiki/compensation-rules`, `/wiki/base-review-fee`, `/wiki/slashing`, `/wiki/finding-rewards`
Function lens	`/wiki/function-lens`, `/features/function-scope`
Match	`/wiki/deployment-match`, `/features/live-code-match`
Scores	`/wiki/trust-scores`, `/protocols/scorer`
M3	`/wiki/m3-scoring`, `/wiki/vaps`, `/wiki/platform-neutrality`
Limitations	`/wiki/limitations`, `/safety`

---

16. Company

Proof of Audits (spaces). Never "VeerSec" in external materials. Never ProofOfAudits as one word.

Founder: MK Veerendra Vamshi, solo, 100% equity.

LinkedIn: https://www.linkedin.com/in/veerendravamshi/

Deck: Adobe Express published deck

Outreach and investor Q&A: `docs/outreach/investor-and-outreach-answers.md`

Locked form answers: `docs/outreach/accelerator-answer-lock.md`

Short brief: `docs/outreach/company-brief.md`

Appendix A. Scoring constants (quick)

M3: severity 15/8/3/0.5/0 · uniqueness 1.00/0.40/0.20/0.10/0.60 · outcome 1.00/0.90/0.70/0.50/0.00 · rank 1.50/1.30/1.15/1.05/1.00 · time 1.00/0.80/0.60/0.45

T4: `0.65R+0.35E` · threshold 150 · hours `max(score/10, 2)`

Reputation: low×1 med×3 high×7 crit×15 missed_basic –10

Slash: crit 35% base / –40 rep · high 20% / –20 · med 10% / –8 · split 60/40 finder/platform

---

Appendix B. Glossary

- **Contest**: time-boxed, pooled review of a locked commit. Our default product.
- **Assigned coverage**: each cluster has an owner. Not a promise that all bugs were found.
- **Duplicate**: substantially the same finding. Not paid as unique work.
- **Cherry-pick**: reviewing only high-EV functions and skipping assigned remainder.
- **Trust Passport**: live proof page: scope, fixes, match, keys, gaps.
- **Native v5 /1400**: native lifecycle score. Separate from External /900.
- **Deployment Match**: live bytecode vs reviewed baseline.
- **Function Review Lens**: wallet action → reviewed function + tier + residual risk.
- **VAPS**: public auditor score 400-1000.

Appendix C. How this paper is structured (and why)

Readers of crypto papers look for: abstract, problem, solution, architecture, economics, roadmap, red flags (Blocklr, ChainClarity, OneKey, 2025-2026 whitepaper guides). Token whitepapers add tokenomics. We have **no token in the beta model**, so that section is omitted on purpose rather than padded.

Order here is the order a buyer should think:

1 What category is this? **Contest platform.**

2. What is broken in that category? **Dups, unowned coverage, cherry-pick.**
3. What is broken after the category's PDF? **User proof.**
4. What do we do? **Assigned contest + living passport.**
5. How, in what order? **Pipeline.**
6. What does it cost, and what is not true yet? **Lock file economics, pre-revenue.**

Proof of Audits: See the proof. Know the risk. Then decide.