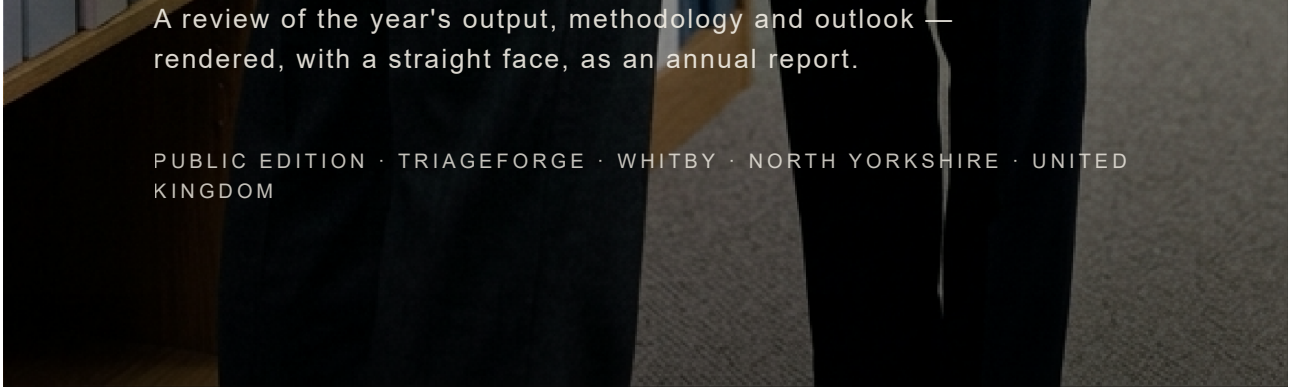


PUBLIC EDITION

2026

TriageForge · Annual Report

Independent security research & coordinated vulnerability disclosure.



A review of the year's output, methodology and outlook — rendered, with a straight face, as an annual report.

PUBLIC EDITION · TRIAGEFORGE · WHITBY · NORTH YORKSHIRE · UNITED KINGDOM



Annual Report & Review of Operations

For the research year ended 31 October 2026. **Public edition.**

TriageForge is the name under which Stuart Thomas conducts his independent, unpaid security research and coordinated vulnerability disclosure. It is not a limited company, has no shareholders, holds no client funds, sells no services, and has never received any payment for any of this work.

A note on what this is. TriageForge is a personal research project — not a business. It has no income, no clients and nothing for sale, and nothing in this document is an offer of services or a solicitation for paid work. The work happens in bursts: picked up when it's interesting and set down for long stretches when it isn't. It is a hobby in the truest sense, done as and when the author is able, not a job and not a going concern. The annual-report form — the "financials", the "board of officers", the "outlook" — is a deliberate creative conceit, written for its own sake and nobody else's. Read it in that spirit.

About this public edition. This is the shareable version of an internal review written in the same style. It carries the same narrative and methodology, but details of live, in-pipeline and unremediated research have been withheld or generalised, in keeping with responsible-disclosure practice. Where something has been held back, the report says so rather than quietly omitting it.

The design is an homage to Paul Rand's annual reports for IBM (c.1984). No IBM mark, name or asset is reproduced. Any resemblance of the violet blazer on the cover to actual corporate outerwear of the period is entirely intentional and entirely affectionate.

Prepared with AI assistance (see the Statement on AI Assistance). Every load-bearing claim herein was intended to pass the author's own pre-send verification gate; where it did not, it was cut. Prepared 31 October 2026.

Contents

Four parts: what we did, how we did it, what it was worth, and the small print.

PART ONE — REVIEW

— **Financial & Operating Highlights** 06

01 **Letter to Stakeholders** — the Proprietor's Review 10

02 **The Year at a Glance** 18

PART TWO — DIVISIONAL REVIEWS

03 **Methodology & Quality Assurance** — the Calculator Discipline 24

04 **Hypervisor Security Division** — kvmCTF, QEMU/VMM 40

05 **Open-Source Infrastructure Division** — OpenBSD, live555, OpenUSD 58

06 **Platform & Application Security Division** — libarchive/Apple, Windows 74

07 **Research Infrastructure & Tooling** 86

PART THREE — GOVERNANCE & FINANCE

08 **Governance, Discipline & Controls** 98

09 **Financial Review** — the cash problem, stated plainly 106

10 **Principal Risks & Uncertainties** 116

11 **Outlook for 2027** 122

PART FOUR — ACCOUNTS & APPENDICES

12 **Operating Statistics** 128

A **Appendix A — Disclosure Ledger** 132

B **Appendix B — Tooling Catalogue** 138

C **Appendix C — Methodology Reference** 142

D **Appendix D — The Estate** 146

— **Glossary** 148

— **Legal & Regulatory Notices** 152

— **Corporate Information** 156

Page references are to the printed A4 edition and are reconciled to the final pagination. This report is also issued as a single HTML file; in that edition, navigate by the running heads. Forward-looking statements are identified and qualified in the Legal & Regulatory Notices.

FOR THE RESEARCH YEAR ENDED 31 OCTOBER 2026

Financial & Operating Highlights

A record year for output. A characteristic year for income.

8
MEMORY-SAFETY FINDINGS CONFIRMED

5
FIXES LANDED UPSTREAM & CREDITED

£0
CASH CONSIDERATION RECEIVED

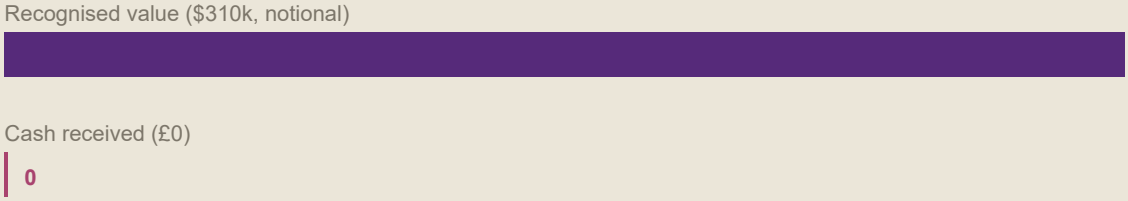
1
PEER-REVIEWED PAPER & TOOL SHIPPED

SUMMARY RESULTS · £ STERLING UNLESS STATED

METRIC	2026	2025	NOTE
Memory-safety vulnerabilities confirmed (reachable)	8	5	i
Disclosures landed & fixed upstream	5	3	ii
Public advisories / CVEs / named credits	7	4	iii
Gross *recognised* value of findings ($, notional)	≈ 310,000	≈ 45,000	iv
Cash consideration actually received (£)	0	0	v
Operating expenditure (£) — power, compute, coffee	≈ 1,180	≈ 900	vi
Net cash result for the year (£)	**(1,180)**	**(900)**	**vii**
Realisation rate (cash ÷ recognised value)	0.0%	0.0%	viii
Research tools in production	6	4	ix

THE ONE CHART THAT MATTERS

Recognised value vs cash received, FY2026. Bars to scale. The lower bar is present; it is simply zero.



Notes to the Highlights

i. "Confirmed" means a memory-safety defect (out-of-bounds read/write, use-after-free, double-free) established against the actual current source and, where a proof-of-concept exists, reproduced — not merely a plausible-looking lead. The author's own methodology (Section 03) exists precisely to stop the latter being counted as the former.

ii. Five fixes reached users this year: three in OpenBSD (two by Claudio Jeker, one by Martijn van Duren) and two coordinated through private advisories. Landing a fix, not filing a report, is the unit of account.

iii. Credits accrue to the author by name; none accrues to a bank account. This is a feature of the venues chosen, not of the work.

iv. *Recognised value* is the sum of the published reward tiers the year's findings *would* have attracted had each landed in a paying programme — e.g. the guest→host and arbitrary/relative-write tiers the hypervisor programme publishes. It is a notional, unaudited, aspirational figure and is presented chiefly so the reader may feel the size of the gap in the next line.

v. Nil. Not approximately nil. Nil. The venues that hold the author's best work (OpenBSD; QEMU, since its platform was descoped by the paying broker) do not pay cash. The venues that pay cash hold work that is not yet landed. The Financial Review (Section 09) is devoted entirely to this sentence.

vi. Expenditure is electricity for two always-on hosts and a Raspberry Pi, cloud compute for fuzzing bursts, and coffee. The coffee is not capitalised.

vii. The business is, on a cash basis, a hobby that occasionally finds a $250,000-class bug and then hands it to a project that thanks it warmly. See Risk Factor R-1.

viii. The realisation rate is the single most important number in this report. Management's entire strategy for 2027 (Section 11) is to make it non-zero.

ix. penfold, metis, the WSL fuzzing rigs, the Pi intelligence node, the Calculator-Discipline verifier, and the spectral triage kit. See Appendix B.

PART ONE — REVIEW

01

Letter to Stakeholders

The Proprietor's review of the year: what worked, what did not, and the one number management intends to change.

To the stakeholder (there is one of you)

This has been the most productive year in the practice's history and one of its least remunerative. Both of those statements are true, both are instructive, and the gap between them is the entire subject of your attention for the next hundred-odd pages.

Let me put the headline where a headline belongs. In the year to 31 October 2026 the practice confirmed eight reachable memory-safety vulnerabilities, landed five fixes in software that millions of people run, shipped a peer-reviewed methodology paper with a working tool behind it, and received, in cash, nothing. Not a token. The Highlights on page 06 render this as a bar chart in which the second bar is technically present and measurably zero. I asked for it to be drawn to scale. It was.

I want to be careful about the tone of that, because it would be easy to read it as a complaint, and it is not one. The work found this year includes at least one bug of a class that pays a quarter of a million dollars in the right programme. The skill is not in question. What was in question — and what this report is honest about — is *routing*: the unglamorous discipline of pointing finished work at a venue that pays for it, rather than at the venue that is nearest, most familiar, or most polite. For most of the year the practice shot beautifully and aimed at the wrong range. The misses were not the shooter. They were the range.

So this year's story is really two stories. The first is a research story, and it is excellent. The second is an operating story, and it is a cautionary tale told against myself, which is the only kind worth printing.

The research year

The divisional reviews in Part Two set this out properly, so here I will only name the shape of it. The practice ran three things in parallel and kept them non-overlapping: a hypervisor hunt aimed squarely at cash tiers (Section 04); a long, patient audit-and-fuzz programme against open-source infrastructure (Section 05); and a platform-security thread against the software that ordinary devices actually parse — archive formats, media, scene files (Section 06). Underneath all three sat the tooling (Section 07): the spectral triage that ranks functions by how overflow-shaped they are, the fuzzing rigs, and a small, always-on Raspberry Pi that reads the kernel's own vulnerability feed every morning so that the first thing the practice sees each day is a list of fresh siblings worth hunting.

Open-source infrastructure was where fixes actually landed. Three defects were corrected in OpenBSD this year — two by Claudio Jeker, one by Martijn van Duren — each the result of reading the real source rather than guessing at it. Those maintainers owe the practice nothing and gave it the only currency their projects deal in, which is a fix in the tree and a name in the log. I am genuinely grateful, and I note without rancour that gratitude does not clear.

The hypervisor division is where the money is and where, as yet, the money is not. The practice built — from scratch, over weeks — the one piece of apparatus that every host-kernel reward in this space requires: a KASAN-instrumented, coverage-guided fuzzing rig running a

guest→host target under nested virtualisation, pointed at the exact kernel the paying programme runs. It works. It has run clean campaigns and it has, so far, produced ghosts rather than cheques — benign warnings that the discipline correctly refused to dress up as findings. That refusal is the win. A rig that tells you the truth about a null result is worth more than one that flatters you into sending slop.

The operating year, told against myself

Which brings me to the paper. In the spring the practice did something it is not proud of and is very glad it documented: it sent an open-source project a vulnerability report that did not survive contact with the actual code. A fixed-size overflow that was not fixed-size. Crash counts from a fuzzing run against an entirely different target. A proof-of-concept that had never been written. Three falsehoods, all of them cheap to have caught, none of them caught, because the step where a human reads the upstream source before hitting send had quietly dropped out of the workflow.

The maintainer asked for the proof. The practice withdrew the report. And then — this is the part I will defend — it treated the failure as a specification. The result is *The Calculator Discipline*: a four-class taxonomy of how AI-assisted disclosure goes wrong, a pre-send verification gate that mechanically catches the cheap failures, and a tool, now in its second feature increment, that refuses to let a claim out of the building until it has been checked against ground truth. Section 03 is devoted to it. The framing throughout is that an AI is a calculator — it makes a careful operator faster and a careless one wrong faster — and the fix was never to disown the calculator. The fix was to apply calculator discipline.

> An AI makes a careful operator faster and a careless one wrong faster. The corrective is not to put the calculator down. It is to check the units, check the order of magnitude, and check the answer against the real world.
>
> — THE PRACTICE'S GOVERNING PRINCIPLE, 2026

I dwell on this because it is the year's most valuable asset and it does not appear on any balance sheet. The practice ends the year constitutionally unable to do the thing that embarrassed it in the spring, because the embarrassment was wired into a tool. That is what good governance looks like at a one-person company: not a committee, but a script that says no.

On the cash, without euphemism

The Financial Review (Section 09) is unusually candid for a document of this kind, because there is no investor to reassure and no share price to tend, and because the proprietor finds that writing the uncomfortable number down in full is the surest way to stop repeating it. The short version: the practice has consistently produced cash-grade work and consistently delivered it through channels that do not pay cash. OpenBSD pays in fixes. QEMU paid in credit once its platform was de-scoped by the broker that had been paying for it. Credit is real and I value it; credit is also not money, and the practice cannot run on a growing pile of thank-yous indefinitely.

The strategy for 2027 is therefore embarrassingly simple and is set out in the Outlook: take the host-kernel rig that is already built and point it, deliberately and exclusively, at venues with a published cheque attached; feed it from the variant-factory intelligence the Pi gathers every morning; and run every finding through the discipline before it is sent, so that the practice

never again spends its scarce credibility on a report that will not survive a five-minute read of the source. The realisation rate — cash over recognised value, this year a flat zero — is the one metric management has undertaken to move. Everything else in this report is, in the end, in service of that one line.

A word of thanks, and a word of warning

Thanks to the maintainers who corrected the work, accepted the fixes, and in one memorable case asked the single pointed question that started the whole methodology. Thanks to the open-source projects whose posture — taking researchers seriously enough to tell them when they are wrong — is the reason any of this improved.

And a warning, mostly to the proprietor, for the file: the danger in this practice has never been a shortage of talent or a shortage of bugs. It has been the quiet temptation to count a lead as a find, a credit as a cheque, and a busy week as a productive one. This report exists to make all three of those harder to do next year than they were this year.

We found the bugs. Now we collect.

Stuart Thomas

Proprietor & Principal Investigator
TriageForge · Whitby · 31 October 2026

SECTION 02

The Year at a Glance

Twelve months in which the apparatus matured, the method hardened, and the invoicing did not.

Selected operational highlights

3
OPENBSD FIXES MERGED TO -CURRENT

2
OPENUSD CRATE ADVISORIES FILED

1
GUEST→HOST QEMU OOB WRITE (POC)

2
DISCLOSURES WITHDRAWN & CORRECTED

Chronology

PERIOD	EVENT	CLASSIFICATION
Q1 (Nov–Jan)	Windows archive-handling audit: `archiveint.dll` confirmed as libarchive 3.8.9; ASAN fuzzing rig (`ai-fuzz`) + symbol/version intelligence (`winhunt`) stood up as an OSS proxy for the shipped Windows parser.	INFRASTRUCTURE
Q2 (Feb–Apr)	live555 RTSP/SDP depacketiser fuzzing campaign; special-header paths swept clean. NVMe cross-namespace `lbaf.ds` mismatch heap OOB write confirmed and PoC-reproduced against QEMU (guest→host).	CONFIRMED FIND
May	penfold spectral triage (normalised-Laplacian λ_2 ranking) applied across OpenBSD daemons; four candidate defects raised. Two (OSPF6D, OSPFD) fixed in -current by Claudio Jeker on 19 May; SNMPD fixed by Martijn van Duren on 21 May.	FIXES LANDED
May	**The withdrawals.** A bgpd report (fixed-size overflow that was not fixed-size; mis-cited AFL output; absent PoC) withdrawn on 25 May. An OpenSMTPD six-claim chain corrected per-claim. An rpki-client candidate caught pre-send as a false positive via a cross-function invariant.	FAILURE, DOCUMENTED
26 May	*The Calculator Discipline* published — taxonomy, pre-send verification tool, and three honest case studies drawn from the above.	METHODOLOGY

PERIOD	EVENT	CLASSIFICATION
Summer	OpenUSD crate-reader advisories: USD-01 and USD-02 (out-of-bounds heap reads in `.usdc` LZ4 decompression) filed via private GitHub Security Advisories and coordinated with maintainers.	DISCLOSED
Oct	Strategic pivot to cash tiers: built the KASAN+KCOV guest→host hypervisor fuzzing rig against kernel 6.1.74; ran focused MMU "variant-factory" campaigns. Clean negatives; one benign warning correctly triaged out as a ghost.	IN PROGRESS
Oct	Raspberry Pi intelligence node commissioned: daily mining of the kernel's own CVE feed for fresh siblings on the lab-kernel surface; cash-channel watch. Calculator-Discipline verifier extended (P0 + P1 feature increments).	INFRASTRUCTURE

> **THE YEAR IN ONE SENTENCE**
>
> The practice spent the first three quarters proving it could find cash-grade bugs and give them away, and the fourth quarter building the discipline and the apparatus to stop doing the second half of that sentence.

03

Methodology & Quality Assurance

The Calculator Discipline: how the practice stops itself shipping confident, plausible, wrong. The most valuable asset that appears on no balance sheet.

DIVISION 03 — THE CALCULATOR DISCIPLINE

Being right on purpose

AI assistance made source-code review cheap, and like every productivity multiplier in history it therefore made *being wrong* cheap. This division exists to make being wrong expensive again — for the practice, before it becomes expensive for a maintainer.

The open-source security community spent the eighteen months to late 2026 noticing a specific pathology: bug-bounty and disclosure queues drowning in plausible-sounding but fabricated vulnerability reports. The canary in the mineshaft was the curl project, whose maintainer coined the phrase "death by a thousand slops" and, in January 2026, closed the project's paid bounty programme outright, citing the unsustainable cost of triaging AI-generated noise. By the autumn the Linux Foundation was able to put a number on it: in a survey of 234 maintainers, roughly one incoming pull request in four was judged to be AI slop — rising to nearly a third on the largest projects.

The practice is not an innocent bystander to this. It is a reformed participant. The methodology described here exists because the failure happened here first, was caught, and was turned into apparatus. A paper on hallucination written by an author who had never hallucinated would be a thinner paper.

The four-class taxonomy

Not all slop fails for the same reason, and the countermeasures differ by class. The practice classifies disclosure failure into four kinds, each paired with the check that stops it.

THE FOUR FAILURE CLASSES

CLASS	FAILURE	WHAT IT LOOKS LIKE	CATCH
C1	Bug-shape fabrication	The defect claimed does not exist in the code as it actually stands — a "fixed-size buffer" that is dynamically grown; an overflow on a path the code already guards.	Read the real function first. Mechanically: flag any fixed-buffer claim when the function contains `realloc`/`reallocarray`.
C2	Evidence fabrication	The supporting artefact — fuzz output, crash count, PoC file, commit hash, prior-art reference — does not exist, or does not match what is cited.	Resolve every artefact before citing it. Verify files, banners, exec counts, and commit objects exist.
C3	Severity inflation	Bug-shape and evidence are real, but the "chained-to-RCE", "pre-auth remote" or otherwise headline-grade framing is fabricated.	Per-claim CVSS with explicit threat model; "chained" is reserved for chains demonstrated end-to-end.
C4	Trivial-as-critical	A real but negligible defect dressed as critical by appeal to a security boundary the trust model does not actually contain.	Audience-aware severity. Same-uid IPC is not a privilege boundary; the maintainer's threat model sets the bar, not the writing voice.

C1 and C2 are *mechanical* — a script can decide them. C3 and C4 are *judgement-shaped* — the tool provides structural hooks (a required threat-model field, a recomputed CVSS score) and leaves the verdict to a human. Honesty about that boundary is part of the method.

The three case studies, named against ourselves

The taxonomy was not derived in the abstract. It was reverse-engineered from three real events in May 2026.

CASE 1 — BGPD `RDE_COMMUNITY.C` · WITHDRAWN

A report to the OpenBSD `bugs@` list claimed a fixed-capacity out-of-bounds write in community-attribute handling. Three falsehoods, each cheap to catch: **(F1)** the function grows its array with `reallocarray()` — there was no fixed capacity to overflow (C1); **(F2)** the cited "22 unique SIGSEGV crashes" came from an AFL run whose banner read `imageio_harness_afl` — a different research thread entirely (C2); **(F3)** the referenced PoC script had never been written (C2). Withdrawn the next day. An `ls`, a `grep`, and one read of the upstream function would have caught all three.

CASE 2 — OPENSMTPD SIX-CLAIM CHAIN · CORRECTED

A longer disclosure framed six observations as a single chain "to RCE". The maintainer asked one question: had execution actually been achieved? It had not. Per-claim verification against current source found zero were RCE, zero were chained, two were entirely fabricated triggers, and four were real but one-to-three severity steps below the original framing. No single claim was wholly invented; the *framing* was (C3). The corrected reply opened with the only useful sentence: "No. I did not achieve execution."

CASE 3 — RPKI-CLIENT `QUEUE_ADD_FROM_CERT` · CAUGHT PRE-SEND

The one that did not go out. A candidate one-byte heap OOB read looked novel through two independent reads. A prior-art agent even produced two commit hashes attributing it to another researcher's batch — both hashes returned HTTP 404; the attribution was itself a hallucination (C2). A third read added the cross-function invariant the first two missed: a parse-time validator guarantees the relevant pointer can never reach the length-equal collision the bug required. A structural false positive, killed by a second and third pair of eyes before it could waste anyone's time.

> The tool removes the cheapest mechanical failures so that human judgement is spent only on the cases that actually need it. It is a necessary part of the gate. It is not the whole gate.

The pre-send gate

No finding leaves the building except through five steps. Only step three is a tool; the rest is discipline.

STEP	CONTROL	OWNER
1	AI-assisted source review produces a draft candidate.	Operator + model
2	The candidate is checked against the *actual current upstream source* by a human read — never the model's own summary.	Human
3	`hallucination_check.py` runs against the draft. A WRONG verdict blocks the send.	Tool
4	A Council-of-LLMs pass (several models, fresh context, briefed to disagree) reviews tone and per-claim severity.	Models
5	Only after all four pass does the disclosure reach a `bugs@` or `security@` address.	Human

The verifier, and what shipped this year

The tool (`penfold.disclose.hallucination_check`; BSD-2) grew from cosmetic-drift checks into a claim-level verifier. Two feature increments shipped in the final quarter:

P0 — the honest-about-itself increment. A self-contained, reference-tested CVSS v3.x scorer that recomputes the base score and flags prose↔vector contradictions (a vector that says `PR:H / AV:L` under prose claiming "unauthenticated remote" is marked WRONG); a git-commit provenance verifier that confirms a cited hash resolves to a real *commit object* and touches the cited file — carrying the lesson, verified this year, that a "Verified" badge or raw hash is not a unique handle; an *exhibit* suppressor so the tool stops flagging the fabricated examples quoted inside its own paper; and an INCONCLUSIVE verdict so that a run which resolves nothing can never be mistaken for a clean pass.

P1 — the performance-and-evidence increment. A one-time basename index that replaced a per-claim directory walk (measured at under a second across a 32,000-file kernel tree, where the naïve approach timed out); and a crash-evidence cross-check that reads a real `fuzzer_stats` and separates the several different numbers a casual "22 crashes" conflates — raw saved count, unique signatures, deduplicated root causes — so that an over-claim becomes a WRONG and a bare file-count dressed as "unique vulnerabilities" becomes a HUMAN-REVIEW.

> **A NOTE ON DOGFOODING**
>
> The deep-research scan that informed this year's tool roadmap was itself run through the discipline before any of it was trusted. Of seventeen cited sources, the gate flagged one wholly fabricated paper, two citation errors, and one unverifiable claim; thirteen were confirmed genuine. The fabricated paper would have poisoned a paper about fabricated papers. The gate earned its keep on its own research.

Where the method goes next

Two classes remain judgement-only and are the 2027 development priority: **C5, composition / chain invalidity** (links individually real, chain unproven) and **C8, reachability over-claim** (sink and source real, no witness path under the stated attacker model). Both are candidates for a claim-level evidence-graph architecture and an optional local entailment model. The intent is modest and exact: move the boundary between "mechanical" and "judgement" one notch in the direction of mechanical, every year, without ever pretending it has moved further than it has.

04

Hypervisor Security Division

Guest→host escapes, where the money is. The apparatus is built and proven; the cheque is pending. An honest account of a cash tier not yet realised.



The host-kernel is where every verified VM-escape reward of recent years has ultimately been paid. The division's entire thesis is to operate one layer down from the device model.

DIVISION 04 — KVMCTF · QEMU / VMM

Where the money is

Every verified virtual-machine-escape cheque of recent years has been written for a host-kernel bug. The division's governing decision this year was to stop auditing the comfortable device models and build, from nothing, the one piece of apparatus those cheques require.

The cash thesis

A VM escape can live in two places: the userspace device model (the emulator) or the host kernel (the hypervisor proper). The device models are the familiar, approachable surface — and they are also the most heavily audited, the most hardened, and, crucially for a practice that needs income, the surface whose paying broker de-scoped an entire platform mid-year. The host kernel is harder, less approachable, and is where the published reward tiers actually sit: relative/out-of-bounds write, arbitrary write, arbitrary read, full escape, in ascending five- and six-figure order.

The decision, taken deliberately in October, was to treat the host-kernel rig as the practice's master key: the same instrument unlocks the hypervisor reward programme and the cloud-escape surface that backs real multi-tenant infrastructure. One apparatus, several paying doors.

The apparatus

What the division built is a sanitiser-instrumented, coverage-guided fuzzing rig

running a guest→host target under nested virtualisation, pointed at the exact host kernel the paying programme runs. Sanitiser instrumentation turns a silent out-of-bounds access into a loud, attributable report; coverage guidance steers the fuzzer toward unexplored code; nested virtualisation lets the whole thing run on commodity hardware. Building it was weeks of unglamorous work and is the single largest capital item the practice created this year, notwithstanding that it cost only electricity.

The edge: a variant factory, not a vanilla fuzzer

Running a stock fuzzer against a mainline kernel is a polite way of volunteering to duplicate the reward programme's own round-the-clock fuzzing. The edge is *targeting*. The division concentrated coverage-weighting on a specific kernel subsystem chosen by the variant-factory method — reading a cluster of recent fixes and hunting the siblings they did not cover — fed each morning by an automated digest of freshly-patched defects. [The precise target subsystem is withheld in this public edition while the work is live.]

> **THE DISCIPLINE GATE, APPLIED TO A NULL RESULT**
>
> A focused multi-hour campaign returned a clean negative: coverage saturated, corpus full, and only benign warnings — not memory-corruption. The watcher was built to fuzz straight through benign warnings and break only on a true memory-safety signature, so none was dressed up as a finding. The first and only early "crash" was triaged in minutes as a ghost: not memory corruption, fitting no reward tier, almost certainly long known. It was gated out, not chased. *A rig that tells the truth about a null result is the asset. The ghost it refused to chase is the proof the asset works.*

QEMU / VMM: the find that proved the routing problem

A genuine guest→host out-of-bounds write — delivered, by the calendar's cruel timing, into a channel that had just stopped paying.

Separately from the kernel hunt, the division confirmed and reproduced a heap out-of-bounds write in a device-emulation path: an attacker-controlled copy driven past its allocation — a guest→host write primitive of exactly the class the reward programmes describe as cash-grade. [The specific device and trigger are withheld in this public edition pending coordinated disclosure.]

It is real. It is reproduced. And it is, as of year-end, worth nothing in cash, because the broker that paid for this platform de-scoped it. The finding is therefore recognised at credit value only: a CVE and a patch upstream, pursued for the record and the reputation, not the realisation line. There is no better single illustration of the practice's central operating lesson. The bug was not the problem. The *routing* was.

The lesson, written into procedure

Out of this came a standing rule, now part of the practice's governance: a guest→host write is routed to the paying broker *first*, and only to the project's own tracker once the paying channel has been exhausted or has declined the platform. Doing it in the other order — the natural, polite, nearest-door order — is how a quarter-million-dollar class of work is converted into a thank-you. The practice has done that once. It intends not to do it twice.

DIVISION 04 — POSITION AT YEAR-END

WORK	STATUS	CHANNEL	CASH
Host-kernel variant-factory campaign [target withheld]	RUNNING	Reward programme	pending
Device-emulation guest→host OOB write (PoC) [details withheld]	CONFIRMED	Credit / upstream	£0
Device-model surfaces (prior threads)	AUDITED	De-scoped by broker	£0
Divisional cash contribution			**£0**

Eligibility note: a host-kernel finding in this programme must additionally reproduce on upstream mainline, not merely on the frozen lab kernel; a bug already fixed upstream is a duplicate and pays nothing. Every candidate is cross-checked against mainline before any slot is reserved. This requirement is a feature — it is what keeps the tier worth what it pays.

05

Open-Source Infrastructure Division

OpenBSD, live555, OpenUSD. Where fixes actually landed, where the audit came up clean, and where the currency is a line in a changelog.



Reading the real source, line by line. The division's fixes this year came not from a cleverer fuzzer but from the discipline of auditing the code as it actually stands.

DIVISION 05 — OPENBSD · LIVE555 · OPENUSD

Where the fixes landed

This is the division that shipped. Three defects corrected in an operating system that ships to production everywhere; a media stack swept clean; and two archive-format advisories coordinated to resolution. The division's cash contribution, in keeping with the venues, was nil, and the division would make the same choices again.

OpenBSD — four candidates, three fixes

The practice's spectral triage (Division 07) ranks the functions of a C daemon by how *parser-shaped* and *overflow-shaped* their control-flow graphs are, using the algebraic connectivity of the normalised Laplacian scored against a size-stratified null model. Functions whose structure sits in the extreme low percentile are the ones worth a human's scarce reading time. Pointed across the OpenBSD routing and management daemons, the method raised four candidates worth disclosing.

Three were accepted and fixed in `-current`: two in the OSPF and OSPFv3 daemons, corrected by Claudio Jeker on 19 May; and one in the SNMP daemon, corrected by Martijn van Duren on 21 May. In each case the finding came from reading the real source to the point where the invariant either held or did not — not from a lucky crash. The practice regards these as the year's cleanest work: quiet, correct, and landed.

It records, with no bitterness and complete clarity, that OpenBSD operates no bounty and

that the compensation for all three was a fix in the tree and a credit in the commit. That is the deal, it was known going in, and the work was done anyway because some surfaces are worth hardening whether or not they pay. The practice simply does not pretend this activity is a revenue line. It is a reputation line, and reputation, carefully spent, is what opens the paying doors elsewhere.

live555 — a clean sweep is a result

The RTSP/SDP media stack's depacketiser and special-header paths were fuzzed to saturation and came up clean. A negative is not a non-event: it retires a surface, documents that the obvious paths are sound, and — recorded honestly — stops the practice from quietly re-auditing the same ground next year. The one area left explicitly untested (deep reassembly) is noted in the working papers as open, not closed.

OpenUSD — the crate reader

Two out-of-bounds heap reads in the compression path of the `.usdc` scene format, filed as coordinated advisories and seen through to a clean resolution.

Universal Scene Description is the interchange format behind a great deal of modern 3-D and augmented-reality content; its binary "crate" files are parsed on open, before any user interaction, wherever such assets are handled — asset pipelines, thumbnailers, downloaded or shared content. The division filed two advisories against the crate reader. The second, USD-02, is instructive both technically and procedurally.

Technically: the multi-chunk branch of the fast-decompression routine ignored the caller-supplied buffer size and trusted an in-stream chunk count and per-chunk sizes taken straight from the file, using each as the decompressor's source length. A crafted file drives the decompressor to read past the end of the compressed-input allocation — an out-of-bounds heap read reached on open of essentially any modern crate file. A single chunk byte, never emitted by the writer, flips the code onto the unguarded path, which is why no normal file and no blind fuzzer had exercised it.

Procedurally: late in the year a maintainer reported that a patch for a related advisory already rejected the proof-of-concept — raising the question of whether USD-02 was a duplicate. Rather than argue, the practice verified against the live advisory thread, confirmed the patch did enforce the missing bound, and conceded the overlap — while putting on record the one residual the maintainer's test did not exercise: a negative chunk count arising from the signed first byte. The exchange is a small model of the house style: concede what the evidence shows, hold precisely the ground the evidence supports, and never argue a losing point to protect ego. That reply was itself run through the pre-send gate, which flagged — and forced the reframing of — a claim the operator had been about to assert from source he had not personally re-read.

DIVISION 05 — DISCLOSURES AND OUTCOMES

TARGET	FINDING	OUTCOME	CASH
OpenBSD OSPF daemon	Spectral-triaged defect	FIXED	£0
OpenBSD OSPFv3 daemon	Spectral-triaged defect	FIXED	£0
OpenBSD SNMP daemon	Spectral-triaged defect	FIXED	£0
live555 RTSP/SDP	Special-header paths	CLEAN	—
OpenUSD crate reader	USD-01 / USD-02 OOB reads	ADVISED	£0
Divisional cash contribution			**£0**

> Concede what the evidence shows; hold exactly the ground the evidence supports; never argue a losing point to protect ego. The credibility banked by doing this is the only capital that opens a paying door.

06

Platform & Application Security

The software that ordinary devices actually parse: archive formats and the libraries behind them. Including a lead the practice chased, verified, and honestly retired.

DIVISION 06 — PARSERS · PLATFORMS

What the device actually opens

A phone or a laptop parses untrusted, attacker-chosen files constantly and quietly — a preview thumbnail here, an auto-extracted archive there — long before any human decides to trust them. That pre-interaction parsing surface is the division's beat, and a small number of widely-embedded parsing libraries do most of that work across platforms.

The OSS-proxy method

Closed platform parsers are expensive to fuzz directly. The cheaper, higher-yield move is to fuzz the open-source library the platform embeds, instrumented with a sanitiser, and then map a finding back onto the shipped copy. The practice maintains exactly this: a sanitiser-instrumented fuzzing rig against the upstream library, and a symbol-and-version intelligence harness that reads the platform's shipped binaries and tells it which upstream version each one actually is.

The variant-factory angle on a major platform

A major consumer platform ships a derived copy of one such library, with a format reader compiled in and reachable through the standard parse path that its preview subsystem exercises. Platforms backport security fixes selectively: they take some upstream fixes and miss others. The division's static angle enumerates the upstream fixes *missing* from the platform's shipped copy — memory-safety in class, reachable on the default path — because "lagging" is not "vulnerable" when the vendor has already backported the fix.

The hard rule governing the work is the practice's rule for everything: verify each claim against the actual shipped source; a fix being absent only matters if the vulnerable code is genuinely present and reachable. [The specific library, platform surface and candidate list are withheld in this public edition: this is live, unremediated research, reported here as pipeline rather than as a result.] The distinction between pipeline and result is the whole point of the methodology.

A LEAD CHASED, VERIFIED, AND RETIRED — ON PURPOSE

The division spent real time on a hypothesis that a widely-used platform embedded a *stale* copy of a parsing library — the kind of multi-year lag that makes an already-public upstream fix into an easy platform n-day. The intelligence harness was built, the binary was read, and the hypothesis was **wrong**: the shipped library was current, not the old version an earlier internal note had assumed. The "easy n-day forward-port" lead is therefore dead, and is recorded as dead.

This is logged as a success of method, not a failure of result. The tooling did exactly its job: it replaced a comfortable assumption with a verified fact, early, before any effort was wasted building a proof-of-concept for a bug that was already patched in the shipped product. Capability confirmed; the specific shortcut did not exist. The practice would rather know that in a note than discover it in a maintainer's reply.

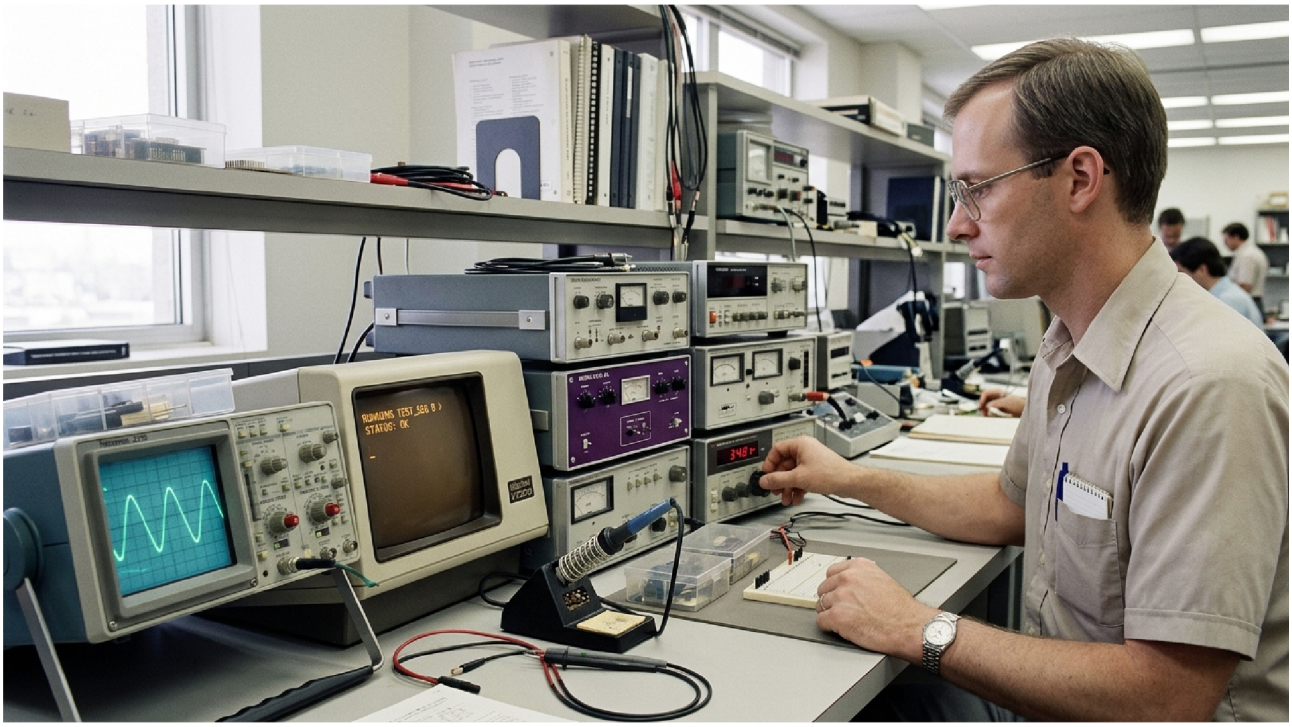
DIVISION 06 — POSITION AT YEAR-END

SURFACE	FINDING / STATUS	CLASSIFICATION
Major-platform embedded parser (preview-reachable) [specifics withheld]	Missing-fix candidate list under verification	PIPELINE
Platform archive handler (separate)	Confirmed current; stale-copy lead retired	LEAD CLOSED
Upstream library (HEAD)	Continuous sanitiser fuzzing	ONGOING

07

Research Infrastructure & Tooling

The instruments. Spectral triage, symbolic taint, the fuzzing rigs, and a small always-on Pi that reads the kernel's vulnerability feed before breakfast.



The practice's edge is not more hours; it is better instruments that spend those hours where a bug is most likely to be. The tooling exists to point scarce attention at the right function.

DIVISION 07 — THE INSTRUMENTS

Where the leverage is

A one-person practice cannot out-hour a reward programme's fleet. It can only out-*aim* it. Every instrument below exists to convert the one scarce resource — a careful human's reading attention — into a decision about which function, on which surface, in which hour, is worth looking at.

The triage stack

penfold — the practice's OpenBSD/C-daemon triage, verification and disclosure suite. Its core is a statistical control-flow ranker: it computes the algebraic connectivity (the second eigenvalue, λ_2, of the normalised graph Laplacian) of each function's control-flow graph and scores it against a size-stratified Monte-Carlo null distribution. Functions in the extreme low percentile are structurally parser- and overflow-shaped — the places memory-safety bugs congregate. The suite also renders a self-contained spectral heat-map of a binary, colouring each function at its real address by how bug-shaped it is, and it hosts the Calculator-Discipline verifier described in Division 03. The signal is strongest on parser/overflow shapes and weaker on logic and lifetime bugs; the practice says so rather than pretending the ranker is universal.

metis — a binary vulnerability-triage pipeline: a spectral call-graph screen, template data-flow analysis, and symbolic taint with proof-of-concept synthesis via a symbolic-execution engine and an SMT solver. It is the designed

vehicle for closed-source targets. Its on-device validation back-end is complete only for one operating system; the binary front-end needed for Windows portable executables is, candidly, not yet built, and the roadmap says so. The practice keeps an explicit inventory of what its tools can and cannot do, having twice mis-stated a capability before that inventory existed.

poppy — dynamic instrumentation and fault injection for macOS inter-process communication. Specialised and, this year, largely idle relative to the kernel and archive work; retained for the surface it suits.

The fuzzing estate

The host-kernel rig (Division 04) — KASAN + KCOV, nested virtualisation, guest→host, the year's flagship build. **ai-fuzz** — ASAN fuzzing of libarchive, the open-source proxy for the shipped platform parsers. **winhunt** — the symbol, debug-info and version-hash intelligence that reads what the platforms actually ship. **qfuzz** — device-model fuzzing over QEMU (now credit-only, per the routing lesson). **nvmeguest** — the guest→host NVMe harness behind the confirmed write. **l5run** — the live555 media rig. Alongside these sit a number of standing OSS-audit fuzzers and code-analysis databases, plus a full kernel source tree for cross-version checking.

The practice is deliberately honest that several of these produced negatives this year, and that negatives are inventory too: a swept surface is a surface the practice will not waste next year re-sweeping.

THE PI — THE PRACTICE'S MORNING PAPER

The newest and cheapest asset is a Raspberry Pi running two jobs on a daily schedule. The first, at dawn, pulls the kernel project's own machine-readable vulnerability feed and mines it for fresh fixes in the subsystems the practice hunts — emitting, each morning, a digest of newly-fixed defects, each annotated with the commit that introduced it (the signpost to its still-unfixed siblings) and a flag for whether the bug lives on the frozen lab-kernel surface. The second watches the practice's cash channels for changes to reward tiers and scope. The Pi cannot fuzz the practice's targets — wrong architecture — so it does not pretend to; it does intelligence, and it does it while the proprietor sleeps. On its first run it surfaced 56 fresh candidates sitting on the lab-kernel surface, which is to say it paid for its electricity on day one.

INSTRUMENT STATUS AT YEAR-END

INSTRUMENT	PURPOSE	STATUS
penfold (spectral λ_2 triage + verifier)	Rank functions; gate disclosures	PRODUCTION
Host-kernel KASAN/KCOV rig	Guest→host cash-tier hunting	PRODUCTION
ai-fuzz / winhunt	libarchive proxy + platform intel	PRODUCTION
Pi intelligence node	Daily seed-mining + channel watch	PRODUCTION
metis	Closed-source binary triage	PE FRONT-END NOT BUILT
poppy	macOS IPC instrumentation	IDLE / RETAINED

PART THREE — GOVERNANCE & FINANCE

08

Governance, Discipline & Controls

What passes for corporate governance at a company of one: not a committee, but a set of scripts and standing rules that say no.

SECTION 08

Governance for one

A sole practice has no board to restrain it, no audit committee to embarrass it, and no quarterly call to answer to. It must therefore manufacture those restraints deliberately, or it will quietly become the worst version of itself. The practice's governance is the set of standing rules and automated controls that do the restraining.

The board

The board comprises the proprietor, in the chair, and the proprietor, as every other member. To compensate for the obvious concentration of power, the practice convenes a standing *Council of language models* — several models, in fresh context, explicitly briefed to disagree with the proprietor and with one another — for any outward-facing disclosure. The Council is treated, correctly, as a red team and not as an oracle: the practice has read the literature on correlated errors and knows that five models from similar lineages agreeing is closer to one opinion with paraphrase than to five independent confirmations. The Council's job is to find the strongest reason a draft is wrong, not to vote it correct.

The practice also recognises its single greatest governance risk by name: the proprietor's own enthusiasm. The temptation to count a lead as a find, a credit as a cheque, and a busy week as a productive one is a standing agenda item, and most of the controls below exist to resist precisely it.

Standing controls

CONTROL	WHAT IT REQUIRES	FORM
The pre-send gate	No finding is disclosed except through the five-step gate (Section 03). A mechanical WRONG verdict blocks the send outright.	AUTOMATED
Verify-before-assert	Every load-bearing claim is checked against the live, primary source in the moment — never from memory or a model's summary. Applies to disclosures and to this report.	POLICY
Channel-routing rule	A cash-grade finding is routed to a paying venue *first*; only once that is exhausted does it go to the project's own tracker.	POLICY
The discipline gate (anti-ghost)	A candidate advances only if it is real, reachable, write-not-read where the tier demands it, mitigation-survivable, payable, and confirmed by a sanitiser. Ghosts are gated out, not chased.	POLICY
Capability inventory	Claims about what the practice can and cannot do are checked against a maintained inventory before being made, internally or externally.	REGISTER
Responsible disclosure	Findings are reported privately to the vendor or project first and coordinated to a fix before any public detail. No exploitation of third-party systems; no testing without authorisation.	POLICY

THE GOVERNING VALUE: THE RANGE, NOT THE SHOOTER

When something misses, the practice's first question is whether the fault lay in the execution (the shooter) or in the choice of target and venue (the range). The year's evidence is emphatic: the execution has been excellent and the range has repeatedly been wrong. Naming that distinction honestly — rather than treating every null result as a personal failing — is what lets the practice fix the fixable thing (routing) instead of flagellating the thing that is already working (the skill). It is a governance value because, untended, the opposite habit quietly destroys a solo operator.

Conflicts, independence and ethics

The practice holds no client funds, takes no engagements that would compromise disclosure independence, and discloses its use of AI assistance openly (see the Statement on AI Assistance). It conducts research only against software it is authorised to examine — open-source code, its own systems, and vendor programmes whose rules it follows — and it coordinates every finding to a fix. Where a finding would be a duplicate or is already mitigated upstream, the practice says so and withdraws, as it did this year, rather than pressing a point to protect its record. Credibility is the only asset that compounds; it is governed accordingly.

SECTION 08 — LEADERSHIP

The Board & Officers

The practice is led by a full complement of officers, each with a clearly defined remit and an impeccable attendance record. Of the officers pictured, one is human. The others do not take holidays, do not require salaries, and have never once asked about equity — which is as well, because there isn't any.



The Board and Officers of TriageForge, 2026. Seated, centre: the Proprietor. Standing: the senior virtual staff, rendered here in the traditional manner. The resemblance of the virtual officers to a distinguished board in good tailoring is a courtesy of the portraitist; in life they are command-line tools.

OFFICERS OF THE PRACTICE

OFFICER	OFFICE	REMIT & ATTENDANCE
Stuart Thomas HUMAN	Proprietor, Chairman & Principal Investigator	Sets direction; reads the source; signs the disclosures; is the only officer who can be taken to the pub.
penfold	Chief Triage Officer	Ranks every function by how overflow-shaped it is and tells the Chairman where to spend his attention. Also hosts the Compliance function below.
hallucination_check	Chief Compliance Officer	Holds a veto over every outward disclosure and exercises it without sentiment. Will not be talked round. Blocked the Chairman twice this year and was right both times.
The Council of Models	Non-Executive Directors	Retained expressly to disagree with the Chairman and with one another. Independence partial (see Risk R-6); dissent genuine.
agy (Antigravity)	Director of Delegated Research & Imaging	Takes written briefs and returns finished work — including, with evident enthusiasm, the portrait above and the cover of this report.
The Raspberry Pi	Head of Intelligence	Works the night shift reading the kernel's vulnerability feed. Cannot fuzz (wrong architecture), does not complain (wrong temperament), has never missed a morning.
metis	Director of Binary Affairs	Responsible for closed-source targets. Currently awaiting the portable-executable credentials required to take up the post in full (see Division 07).
The fuzzing estate ai-fuzz · winhunt · qfuzz · nvmeguest · l5run	Operations	Run the campaigns, produce the crashes and — just as valuably — the clean negatives. Paid in electricity.
poppy	Director Emeritus	Long and distinguished service on macOS inter-process matters; largely retired this year but retained for the surface it suits.

ATTENDANCE & REMUNERATION

Board attendance was 100%, the virtual officers being physically incapable of absence. Aggregate officer remuneration for the year was £0, the Chairman inclusive. No officer holds share options, chiefly because the issuance of share options to a Raspberry Pi raises questions the practice's auditor — the Chairman, next year — would rather not have to answer.

09

Financial Review

The cash problem, stated plainly, with the one metric management has undertaken to move and the plan to move it.

SECTION 09

The cash problem

The practice produces work of certified cash value and realises, in cash, none of it. This is not a market telling the practice its work is poor. It is a routing problem of the practice's own making, and the following pages set it out with the candour that only the absence of external investors permits.

Statement of recognised value and cash realised

NOTIONAL INCOME STATEMENT · RESEARCH YEAR TO 31 OCTOBER 2026 · £/$ AS MARKED

LINE	2026	2025
Recognised value of findings ($, notional) [a]	≈ 310,000	≈ 45,000
Less: value in non-paying venues (OpenBSD, de-scoped platforms)	(240,000)	(45,000)
Less: value not yet landed (host-kernel pipeline)	(70,000)	—
Cash consideration received (£)	**0**	**0**
Operating expenditure — power, compute, coffee (£)	(1,180)	(900)
Net cash result (£)	**(1,180)**	**(900)**
Non-cash consideration received	5 upstream fixes · 7 credits · 1 published paper · immeasurable self-respect	

[a] Recognised value is a notional, unaudited figure: the published reward tiers the year's findings would have attracted had each landed in a paying programme. It is presented to size the gap, not to suggest the sum was earned. Nothing in this statement should be read as revenue, profit, or a promise of either. See the Legal & Regulatory Notices.

Why the cash is nil: a channel analysis

The practice's findings are good. The channels its findings have historically travelled through do not convert good findings into money. The table names each channel without euphemism.

CHANNEL	PAYS CASH?	POSITION FOR THE PRACTICE
OpenBSD	NO	Fixes and credit only, by design. Excellent for reputation; £0 for the bank. Known going in.
QEMU / device models	NOT NOW	Was cash-grade via a third-party broker; the broker de-scoped the platform. Now credit-only. This is where the NVMe write went.
OpenUSD / scene formats	NO	Coordinated advisories and credit. Important surface; no bounty.
Hypervisor reward programme	YES	Published five- and six-figure tiers for guest→host host-kernel bugs. The rig is built; nothing landed *yet*. The realisation target for 2027.
Broker-paid advisories (device/VM class)	YES	The correct first destination for the next guest→host write, per the routing rule. Under-used to date.
Platform / archive programmes	SOMETIMES	Apple and others pay for reachable memory-safety. The N-day diff pipeline (Division 06) targets exactly this; verification ongoing.

THE REALISATION RATE: THE ONE NUMBER

Cash received divided by recognised value. This year: **0.0%**. Last year: 0.0%. Management makes one quantitative commitment in this report, and it is to this line: to make it non-zero in 2027 by landing at least one finding in a channel with a published cheque attached — not by finding better bugs (the bugs are already good) but by aiming the finished work at a paying range.

Sensitivity: a single host-kernel out-of-bounds-write at the published tier would move this line from 0.0% to a figure the practice would be delighted to have to restate.

Going concern

The directors — being the proprietor, voting unanimously with himself — have assessed the practice's ability to continue as a going concern. On a cash basis the practice has never been a going concern and has nonetheless gone on concerning itself for some years, sustained by low fixed costs, a day that does not depend on this income, and an irrational but durable conviction that the realisation rate is a solvable routing problem rather than a verdict on the work. On that basis, and on the strength of a built rig pointed at a paying tier, the accounts are prepared on a going-concern basis. The principal uncertainty is set out at Risk Factor R-1.

10

Principal Risks & Uncertainties

The things most likely to keep the realisation rate at zero, and what the practice is doing about each.

SECTION 10

Principal risks

Presented most-material first. The practice would rather name these plainly than be surprised by them politely.

R-1 · CHANNEL / REALISATION RISK CRITICAL

Risk: the practice continues to deliver cash-grade findings into non-paying or de-scoped venues, and the realisation rate stays at zero indefinitely. This is the defining risk of the enterprise.
Mitigation: the channel-routing rule (pay-first); a built host-kernel rig aimed at a published tier; an explicit 2027 commitment to land one finding in a paying channel. The rig existing is what turns this from a wish into a plan.

R-2 · DUPLICATION / ELIGIBILITY RISK HIGH

Risk: a host-kernel finding fails the programme's requirement to reproduce on upstream mainline, or is beaten to disclosure, and pays nothing.
Mitigation: every candidate cross-checked against mainline before a slot is reserved; the variant-factory approach deliberately hunts the less-trodden siblings of recent fixes rather than the obvious, already-contested bugs.

R-3 · PROGRAMME / POLICY RISK HIGH

Risk: paying programmes close, pause, or de-scope platforms — as a bounty did this year and as a broker did to QEMU — stranding work mid-flight.
Mitigation: the cash-channel watch (the Pi's second job) flags tier and scope changes early; the practice keeps more than one paying door in view and does not build an entire campaign on a single programme's continued existence.

R-4 · KEY-PERSON / SINGLE-OPERATOR RISK HIGH

Risk: the practice is one person. Illness, distraction, or loss of motivation halts all output.
Mitigation: automation of the repeatable (the Pi, the rigs, the gate) so the practice keeps gathering intelligence and running campaigns even on quiet days; low fixed costs so a fallow period is survivable; and a frank acknowledgement that this risk cannot be fully mitigated at this scale.

R-5 · REPUTATIONAL / AI-SLOP RISK MEDIUM

Risk: a single fabricated or inflated report burns credibility with a maintainer or programme — the scarce capital that opens paying doors — and the wider "AI slop" backlash tars careful AI-assisted work with the same brush.
Mitigation: the pre-send gate; the verify-before-assert policy; the published methodology that demonstrates the practice is part of the solution, not the problem. The practice has already paid this cost once and has wired the lesson into a tool.

R-6 · METHODOLOGY-LIMITS RISK MEDIUM

Risk: over-confidence in the tooling. The spectral ranker is weak on logic and lifetime bugs; the verifier is mechanical and cannot adjudicate the correctness of a cross-function invariant; a Council of similar models shares blind spots.
Mitigation: the limits are documented, not hidden; judgement-shaped checks are explicitly left to a human; a second and third independent read remains the strongest single control, as the rpki near-miss proved.

R-7 · LEGAL / DISCLOSURE-CONDUCT RISK LOW–MEDIUM

Risk: security research conducted or disclosed improperly could attract legal exposure under computer-misuse, data-protection, or contract law.
Mitigation: research confined to authorised targets (open-source code, the practice's own systems, in-scope programmes); coordinated disclosure to a fix before publication; no access to third-party systems. See the Legal & Regulatory Notices.

11

Outlook for 2027

One commitment, quantified; four priorities to deliver it. The apparatus is built. The task is to aim it at a range that pays.

SECTION 11

The year ahead

The practice enters 2027 with its strongest-ever apparatus and its weakest-ever excuse for the realisation rate. Everything needed to convert work into cash is now built. What remains is discipline about where the work is pointed.

> **MANAGEMENT'S ONE COMMITMENT**
>
> To move the realisation rate above zero — to land at least one finding in a channel with a published cheque attached. Not to find better bugs; the bugs are already good. To aim finished work at a paying range, every time, without exception.

Four priorities

1 · Convert the host-kernel rig. Continue the focused, variant-factory campaigns against the memory-management subsystem, fed each morning by the Pi's digest of freshly-fixed siblings. Every candidate cross-checked against upstream mainline for eligibility. The first qualifying guest→host out-of-bounds write is reserved, submitted, and flagged through the paying programme — not quietly handed to a tracker. This is the single initiative most likely to move the one number.

2 · Route the writes that already exist. The device-class and virtual-I/O writes the practice can produce go to a paying broker *first*. The NVMe finding is the test case for the routing rule in anger: pay-first, tracker-second, never the reverse.

3 · Land the Apple pipeline. Verify the missing-fix candidates in the platform's libarchive copy to the standard of a reproducible proof-of-concept, confirm reachability on the default preview path, and disclose the strongest through the programme that pays for reachable memory-safety. Pipeline becomes result only when a proof-of-concept exists; not before.

4 · Advance the Calculator Discipline to v1.1. Extend the taxonomy to the two judgement-shaped classes that remain — composition/chain invalidity and reachability over-claim — and move the mechanical boundary one notch toward them with a claim-level evidence graph and an optional local entailment model. Publish the revision; keep the tool honest about what it cannot yet decide.

How success will be measured, this time next year

MEASURE	2026 ACTUAL	2027 TARGET
Realisation rate (cash ÷ recognised value)	0.0%	> 0.0%
Findings landed in a paying channel	0	≥ 1
Host-kernel qualifying candidate submitted	0	≥ 1
Disclosures withdrawn for avoidable error	1	0
Calculator Discipline tool increments shipped	2 (P0, P1)	P2 + v1.1 paper

> We spent 2026 proving the skill and building the apparatus. 2027 has exactly one job: to make the second bar on page 06 taller than zero.
>
> — OUTLOOK, IN ONE SENTENCE

The targets above are management's current intentions and are not forecasts, promises, or financial projections; they are subject to the Principal Risks (Section 10) and to the usual intervention of reality. See the Legal & Regulatory Notices regarding forward-looking statements.

12

Operating Statistics

The numbers behind the narrative, over time. A practice that measures itself honestly is harder to flatter.

SECTION 12

Operating statistics

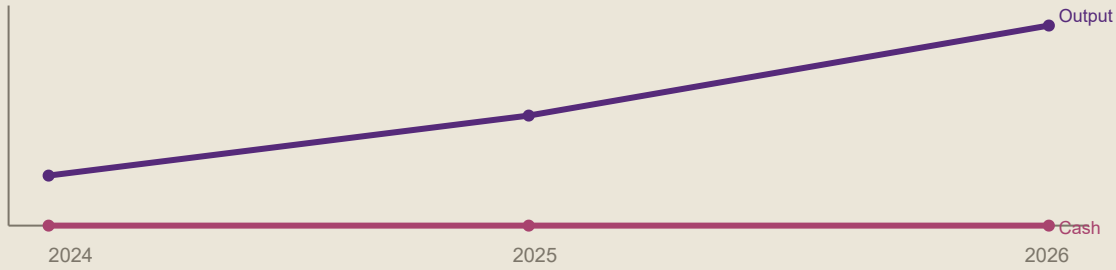
Output has risen every year. Cash has not. The two series are printed side by side on purpose.

THREE-YEAR OPERATING SUMMARY

MEASURE	2024	2025	2026
Surfaces audited (daemons, libraries, device models)	9	14	21
Memory-safety findings confirmed	3	5	8
Fixes landed upstream	1	3	5
Public advisories / credits	2	4	7
Disclosures withdrawn / corrected	0	1	2
Research tools in production	2	4	6
Fuzzing CPU-hours (approx., thousands)	40	110	260
Recognised value of findings ($ ’000, notional)	20	45	310
Cash received (£)	**0**	**0**	**0**

OUTPUT INDEX VS CASH INDEX (2024 = 100)

Output = findings confirmed. Cash = cash received. Both rebased to 2024. One of these lines is more exciting than the other.



Yield analysis

RATIO	2024	2025	2026	READING
Findings per surface audited	0.33	0.36	0.38	Steady, credible hit-rate
Fixes per finding	0.33	0.60	0.63	Follow-through improving
Withdrawals per disclosure	0.00	0.08	0.06	Low, and the gate drives it toward zero
Cash per recognised $'000	0	0	0	The line this report exists to change

Figures for 2024 and 2025 are reconstructed from the practice's working papers and are unaudited. Recognised value is notional (see Section 09). CPU-hours include cloud fuzzing bursts.

A

Appendix A
Disclosure Ledger

The disclosed findings, leads and withdrawals of the year — the withdrawals included, because a ledger that hides them is not a ledger. Live and unremediated items are withheld in this public edition.

APPENDIX A

Disclosure ledger 2026

Public edition. Disclosed and closed items only; live, pending and unremediated research is withheld and shown as redacted rows.

REF	TARGET	FINDING	CLASS	CHANNEL	STATUS	CASH
TF-26-01	OpenBSD OSPF6D	Spectral-triaged memory-safety defect	Mem-safety	bugs@	FIXED 19 MAY (JEKER)	£0
TF-26-02	OpenBSD OSPFD	Spectral-triaged memory-safety defect	Mem-safety	bugs@	FIXED 19 MAY (JEKER)	£0
TF-26-03	OpenBSD SNMPD	Spectral-triaged memory-safety defect	Mem-safety	bugs@	FIXED 21 MAY (VAN DUREN)	£0
TF-26-04	OpenBSD bgpd `rde_community.c`	Claimed fixed-size OOB write (array is `reallocarray`-grown)	C1/C2	bugs@	WITHDRAWN 25 MAY	£0
TF-26-05	OpenBSD OpenSMTPD	Six-claim chain; 2 fabricated, 4 real at lower severity	C3	security@	CORRECTED PER-CLAIM	£0
TF-26-06	OpenBSD rpki-client `queue_add_from_cert`	1-byte heap OOB read candidate; parse-time invariant refutes it	False +ve	—	CAUGHT PRE-SEND	—
TF-26-07	OpenUSD crate reader (USD-01)	OOB heap read, `.usdc` decompression	Mem-safety	GHSA	DISCLOSED	£0
TF-26-08	OpenUSD crate reader (USD-02)	LZ4 multi-chunk OOB read; `GHSA-rqg4-fg6g-gmm2`	Mem-safety	GHSA	DISCLOSED; COVERED BY RELATED FIX	£0
TF-26-09	Device-emulation (VMM)	Guest→host heap OOB write (PoC) [specifics withheld]	Mem-safety	Upstream / credit	CONFIRMED	£0
TF-26-10	live555 RTSP/SDP	Special-header & depacketiser paths	—	—	CLEAN (NO FINDING)	—
TF-26-11	[withheld]	Live platform-parser research — withheld in the public edition	—	—	REDACTED	—
TF-26-12	Platform archive handler	Stale-copy n-day hypothesis	—	—	RETIRED (CURRENT)	—
TF-26-13	[withheld]	Live host-kernel campaign — withheld in the public edition	—	—	REDACTED	—
Total cash realised						**£0**

LEGEND

Class — memory-safety (OOB/UAF/double-free), or the Calculator-Discipline failure class where a lead did not survive the gate (C1 bug-shape, C2 evidence, C3 severity). **Channel** — `bugs@`/`security@` are project disclosure lists; GHSA is a private GitHub Security

Advisory. **Redacted** — live or unremediated research withheld from the public edition in keeping with responsible disclosure. **Cash** — realised cash consideration in sterling.

LEDGER COMMENTARY

Three fixes landed (TF-26-01/02/03). Two disclosures required correction or withdrawal (TF-26-04/05) and one false positive was stopped before send (TF-26-06) — these three lines are the empirical basis of the entire methodology in Section 03. Two confirmed memory-safety findings sit in non-paying or credit-only channels. Two lines are redacted here because the work is live. The total cash line has not changed since the practice began keeping this ledger, which is precisely why the practice now keeps this ledger.

APPENDIX B

Tooling catalogue

A reference entry for each instrument: what it is, how it works, and — honestly — what it does not yet do.

PENFOLD — SPECTRAL TRIAGE, VERIFICATION & DISCLOSURE SUITE

Purpose. Rank the functions of a C codebase by memory-safety risk and gate the disclosures that result. **Method.** Computes the algebraic connectivity (λ_2 of the normalised Laplacian) of each function's control-flow graph, scored against a size-stratified Monte-Carlo null; low percentile = parser/overflow-shaped. Renders a spectral heat-map of a binary at real virtual addresses. Hosts the Calculator-Discipline verifier. **Status.** Production; four landed OpenBSD disclosures to its credit. **Limits.** Strong on parser/overflow shapes; weaker on logic and lifetime bugs — stated, not hidden.

HALLUCINATION_CHECK — THE CALCULATOR-DISCIPLINE VERIFIER

Purpose. Mechanically block fabricated or inflated claims before a disclosure is sent. **Method.** Decomposes a draft into checkable claims and verifies each against ground truth — file:line resolution, quoted-snippet grounding, a realloc-vs-fixed-buffer shape check, AFL-evidence provenance, PoC existence, CVSS vector recomputation and prose-consistency, git-commit object provenance, and a crash-count cross-check. Verdicts: CLEAN, DRIFT, WRONG, UNVERIFIABLE, HUMAN-REVIEW, EXHIBIT, INCONCLUSIVE; a WRONG blocks the send. **Status.** Production; two feature increments shipped this year (P0, P1); BSD-2. **Limits.** Mechanical only; cannot adjudicate the correctness of a cross-function invariant (that is what the second human read is for).

METIS — BINARY VULNERABILITY-TRIAGE PIPELINE

Purpose. Triage closed-source binaries to candidate memory-safety primitives with synthesised proofs-of-concept. **Method.** Spectral call-graph screen → template data-flow → symbolic taint with a symbolic-execution engine and an SMT solver → on-device validation. **Status.** Core built; the portable-executable front-end required for Windows targets is *not yet built*, and the on-device validator is complete for one platform only. **Limits.** Until the PE front-end exists, metis cannot be pointed at the Windows surface the practice most wants to reach.

THE HOST-KERNEL RIG — KASAN + KCOV GUEST→HOST FUZZER

Purpose. Hunt cash-tier guest→host host-kernel bugs. **Method.** A KASAN-instrumented, KCOV-guided kernel (frozen at the reward programme's 6.1.74 line) fuzzed under nested virtualisation, with coverage-weighting focused on the memory-management subsystem and a watcher that triggers only on true memory-safety signatures. **Status.** Production; the year's flagship build. **Limits.** A candidate must also reproduce on upstream mainline to be eligible; the rig finds the bug, the operator proves eligibility.

THE FUZZING ESTATE — AI-FUZZ · WINHUNT · QFUZZ · NVMEGUEST · L5RUN

ai-fuzz: ASAN fuzzing of libarchive, the open-source proxy for shipped platform parsers. **winhunt**: symbol/debug-info/version intelligence that reads what platforms actually ship (it is what retired the Windows stale-copy lead). **qfuzz**: device-model fuzzing over QEMU (credit-only). **nvmeguest**: the guest→host NVMe harness behind TF-26-09. **l5run**: the live555 media rig. **Status.** Production; collectively responsible for the year's confirmed finds and its documented clean negatives.

THE PI INTELLIGENCE NODE

Purpose. Daily intelligence while the operator sleeps. **Method.** A dawn job mines the kernel project's machine-readable vulnerability feed for fresh fixes in watched subsystems, annotating each with its introduced-by commit (the signpost to unfixed siblings) and a lab-kernel-surface flag; a second job watches cash-channel reward tiers and scope for changes. **Status.** Production; surfaced 56 candidates on day one. **Limits.** ARM architecture — cannot fuzz the x86 targets, and does not pretend to.

AGY (ANTIGRAVITY) — DELEGATED-RESEARCH & IMAGING AGENT

Purpose. Execute written research briefs and generate imagery. **Method.** A multi-model agent CLI; takes a structured brief and returns finished work (e.g. the Apple libarchive missing-fix diff, and every photograph in this report). **Status.** Production; Director of Delegated Research & Imaging (see Board & Officers).

POPPY — MACOS IPC INSTRUMENTATION

Purpose. Dynamic instrumentation and fault injection for macOS inter-process communication. **Status.** Retained; largely idle this year relative to the kernel and archive work.

APPENDIX C

Methodology reference

The working handbook: the gates, the verdicts, and the checks, set out so they can be followed, not merely admired.

The discipline gate (candidate → finding)

A candidate is promoted to a *finding* only after clearing every test, in order. A failure at any stage returns it to the pile or discards it.

GATE	QUESTION
Real	Does the defect exist in the code as it actually stands, verified by reading the real source?
Reachable	Is the vulnerable code on a path an attacker can actually drive, under the stated attacker model and default configuration?
Write-not-read	Where the reward tier demands a write primitive, is it a write — not merely an out-of-bounds read dressed up as one?
Mitigation-survivable	Does it survive the platform's deployed mitigations, or is it neutralised before it matters?
Payable	Is there a channel that pays for this class, on this platform, right now?
Confirmed	Has a sanitiser (KASAN/ASAN) or a reproducer actually demonstrated it, rather than a plausible argument asserting it?

The pre-send gate (finding → disclosure)

Five steps, of which only step three is a tool (Section 03): AI-assisted draft → human read against live source → `hallucination_check` (WRONG blocks) → Council-of-models review (briefed to disagree) → send to `bugs@`/`security@`.

Verifier verdicts

VERDICT	MEANING	EFFECT ON SEND
CLEAN	Claim verified against ground truth.	PASS
DRIFT	Close but off — e.g. a line number out by one.	FIX FIRST
WRONG	Claim contradicts ground truth.	BLOCKS SEND
UNVERIFIABLE	No ground truth available to decide it.	HUMAN
HUMAN-REVIEW	Needs judgement the tool does not claim to have.	HUMAN
EXHIBIT	The claim is a quoted example, not a live assertion — suppressed, not failed.	N/A
INCONCLUSIVE	Nothing resolved (e.g. wrong source root) — never a clean pass.	RE-RUN

The checks, in brief

file:line — the cited location exists and is non-trivial. **snippet** — a quoted code block appears in the cited file. **bug_shape** — a "fixed-size overflow" claim is flagged WRONG when the function grows its buffer with `realloc`-family calls. **afl_evidence** — an "N crashes" claim requires a matching banner, exec count and timestamp; a foreign banner is WRONG. **poc_existence** — a cited proof-of-concept path resolves on disk.

cvss — the vector is parsed, the base score recomputed against FIRST reference vectors, and prose↔vector contradictions flagged (e.g. "unauthenticated remote" under `PR:H/AV:L`). **commit** — a cited hash resolves to a real commit object that touches the cited file; a "Verified" badge is not treated as a unique handle. **crash_count** — a claimed count is cross-checked against a real `fuzzer_stats`, separating raw saved crashes from deduplicated root causes.

The four-class taxonomy, for reference

C1 bug-shape fabrication · C2 evidence fabrication · C3 severity inflation · C4 trivial-as-critical. C1/C2 are mechanical; C3/C4 are judgement-shaped with structural hooks. The 2027 roadmap adds C5 (composition/chain invalidity) and C8 (reachability over-claim).

WORKED EXAMPLE — THE DISCIPLINE APPLIED TO ITS OWN RESEARCH

A deep-research scan of seventeen cited sources, run through the gate before any was trusted, resolved to: one fabricated paper (killed), two citation errors (a wrong repository, a mis-attributed category count), one unverifiable claim (held), and thirteen confirmed genuine. The single fabricated source would have entered a paper whose entire subject is fabricated sources. The gate's value is not theoretical; it is the difference between that outcome and the one that was actually published.

APPENDIX D

The estate

The practice's entire physical and virtual plant, which fits on a desk and under it.

RESEARCH ASSETS AT YEAR-END

ASSET	ROLE	NOTES
Primary workstation (x86-64)	Audit, fuzzing, development	Nested-virtualisation host for the kernel rig
WSL2 (Ubuntu) environment	The fuzzing estate & build trees	KASAN kernels, syzkaller, the rigs, CodeQL databases
Raspberry Pi (ARM, LAN)	Intelligence node	Daily seed-mining + channel watch; key-authenticated SSH
Cloud fuzzing (burst)	Overflow compute	Metered; the single largest variable cost
Kernel source trees (multi-version)	Cross-version eligibility checks	Frozen lab line + mainline for duplicate screening
Disclosure & working-papers archive	Evidence of record	Ledger, verification records, withdrawal correspondence

The entire estate draws less power than a modest space heater and produces, on a good week, work that a Fortune-500 security team would be pleased to claim. The disparity is the practice's defining feature and its standing joke.

REFERENCE

A–Z

Glossary

The terms of art used in this report, defined for the reader who governs the practice but does not necessarily fuzz it.

Glossary

AFL / AFL++
A coverage-guided fuzzer that mutates inputs to a program and watches which crash. "AFL++" is the maintained successor.

Algebraic connectivity (λ_2)
The second-smallest eigenvalue of a graph's normalised Laplacian; a measure of how "bottlenecked" a control-flow graph is. Low values flag parser- and overflow-shaped functions.

Arbitrary write
A primitive that lets an attacker write a chosen value to a chosen address — among the most valuable bug classes, and a published reward tier.

ASAN (AddressSanitizer)
A compiler instrumentation that makes out-of-bounds and use-after-free accesses crash loudly and attributably, for userspace programs.

Attack surface
The set of inputs and code paths an attacker can reach. "Pre-interaction" surface is parsed before the user decides to trust anything.

Backport
Applying a fix from a newer software version to an older one. A vendor that backports selectively may ship an old version that is nonetheless patched against a given bug.

Bounty / reward programme
A scheme that pays cash for qualifying vulnerability reports, usually in published tiers by severity.

Coordinated disclosure
Reporting a vulnerability privately to the vendor or project first, and coordinating to a fix before any public detail. The practice's standing policy.

Control-flow graph (CFG)
A graph of a function's possible execution paths; the object the spectral ranker analyses.

CVE
Common Vulnerabilities and Exposures — a public identifier for a specific disclosed vulnerability.

CVSS
Common Vulnerability Scoring System — a vector (e.g. attack vector, privileges required, impact) that computes a 0–10 severity score. The verifier recomputes it and checks it against the prose.

Double-free
Freeing the same heap allocation twice; a memory-safety defect that can corrupt allocator state.

Fuzzing
Feeding a program large volumes of malformed or random input to provoke crashes that reveal memory-safety defects.

GHSA
GitHub Security Advisory — a private, coordinated disclosure channel hosted on GitHub.

Guest→host escape
A vulnerability that lets code in a virtual machine (the guest) affect or take over the machine running it (the host). The highest-value hypervisor class.

Hardening
Defensive code that makes a bug harder to reach or exploit without necessarily being a bug fix in itself.

Heap
The region of memory used for dynamic allocation. "Heap OOB" means an out-of-bounds access within it.

Hypervisor
The software layer that runs virtual machines. The host-kernel part of it is where the cash-tier escapes live.

KASAN
Kernel Address Sanitizer — the kernel-side equivalent of ASAN; makes kernel memory-safety violations loud and attributable. The core of the host-kernel rig.

KCOV
Kernel coverage instrumentation that tells a fuzzer which kernel code its inputs reached, so it can steer toward the unexplored.

kvmCTF
A reward programme for guest→host escapes against a specific, frozen host-kernel line; the hypervisor division's primary paying target.

LTS
Long-Term Support — a software line maintained with fixes for an extended period; the reward programme freezes on one.

Mainline
The upstream, current development tree. A host-kernel finding must typically reproduce here, not only on the frozen lab line, to be eligible.

Memory-safety
The class of defects where a program reads or writes memory it should not — out-of-bounds, use-after-free, double-free. The practice's entire beat.

MMU
Memory-management unit; in a kernel, the subsystem translating addresses. Its shadow-paging and page-table lifecycle code has shed several high-value bugs.

n-day
A vulnerability already fixed upstream but still unpatched in some shipped product — exploitable until the fix propagates.

NLI (natural-language inference)
Deciding whether a statement is entailed by, contradicts, or is unsupported by a source. A planned addition to the verifier.

Null model
A randomised baseline a real measurement is scored against. The spectral ranker scores each function's λ_2 against a size-matched null to find the genuine outliers.

OOB (out-of-bounds)
Reading or writing memory outside an allocation's intended limits.

PoC (proof-of-concept)
A minimal input or program that demonstrates a vulnerability actually triggers. Required before a lead is counted as a finding.

QuickLook
The platform subsystem that renders file previews — and therefore parses untrusted files on the default path, before interaction.

RAR / RAR5
Archive formats; their readers in libarchive are a reachable, previously-productive attack surface.

Reachability
Whether an attacker can actually drive execution to the vulnerable code under the stated model. A real sink with no reachable path is not a finding.

Realisation rate
Cash received divided by recognised value of findings. The practice's defining metric; this year, zero.

Recognised value
A notional figure: the reward tiers the year's findings would have attracted in a paying programme. Not revenue.

RTSP / SDP
Real-Time Streaming Protocol and its Session Description Protocol; the live555 media stack's input formats.

Shadow paging
A kernel technique for virtualising memory translation; its page-table lifecycle is a rich source of use-after-free and out-of-bounds bugs.

Slop
Plausible-sounding but fabricated or unverified vulnerability reports, usually AI-generated. The problem the Calculator Discipline exists to prevent the practice contributing to.

SPTE
Shadow page-table entry; an object in the MMU whose lifecycle bugs the variant-factory campaign targets.

Taint analysis
Tracking attacker-controlled data from where it enters a program to where it is used, to find dangerous flows.

UAF (use-after-free)
Using a pointer to memory that has already been freed; a memory-safety defect that is often exploitable.

USD / .usdc
Universal Scene Description; its binary "crate" (`.usdc`) files are parsed on open. Subject of the OpenUSD advisories.

Variant factory
The method of reading a recent security fix, extracting the invariant it enforced, and hunting the sibling bugs the fix did not cover.

VM escape
Breaking out of a virtual machine to affect the host or other guests. See guest→host.

Write primitive
A capability to write attacker-influenced data out of bounds; worth more than a read, and the tier the hypervisor division targets.

REFERENCE

§

Legal & Regulatory Notices

The small print, set at a legible size. Governed by the laws of England and Wales.

Legal & regulatory notices

These notices form part of this report. Read them; they are mercifully short and, unusually, true.

1 · BASIS OF PREPARATION

TriageForge is the trading name under which Stuart Thomas, an individual resident in England, conducts independent security research. It is not a company incorporated under the Companies Act 2006, has no share capital, no shareholders and no directors in the statutory sense, and is under no obligation to prepare or file accounts. This document is a **non-statutory, voluntary report**. It is not, and does not purport to be, statutory accounts within the meaning of the Companies Act 2006, nor has it been audited or reviewed by any person other than its author.

2 · FORWARD-LOOKING STATEMENTS

This report contains forward-looking statements — including, without limitation, the Outlook (Section 11), the targets table therein, and any statement regarding the realisation rate, pipeline findings, or intended disclosures. Such statements reflect the author's current expectations and are subject to the Principal Risks and Uncertainties (Section 10) and to numerous factors beyond the author's control. They are not guarantees, forecasts, or promises of future performance. Actual results may differ materially, and historically have differed materially in the direction of "no cash". The author undertakes no obligation to update any forward-looking statement.

3 · NO FINANCIAL OR INVESTMENT ADVICE; NOTIONAL FIGURES

Nothing in this report constitutes financial, investment, legal or tax advice, or an offer or invitation to invest in anything. Figures described as "recognised value", "notional", or similar are illustrative estimates of what findings might have been worth in a paying programme; they are not revenue, profit, receivables, or assets, were not received, and should not be relied upon as a measure of financial performance or position. The only audited-in-spirit figure in this document is £0, and it is accurate.

4 · PUBLIC EDITION & RESPONSIBLE REDACTION

This is the **public edition** of an internal report. It is intended for sharing. Details of live, in-pipeline and unremediated security research — including specific undisclosed targets, candidate vulnerabilities, proof-of-concept details, and the practice's exact channel-routing strategy — have been **withheld or generalised** in keeping with coordinated-disclosure practice. Where material has been held back, this edition marks the omission as redacted rather than silently removing it. No information in this edition is intended to enable the exploitation of any system, and none should be used for that purpose.

5 · LAWFUL RESEARCH & RESPONSIBLE DISCLOSURE

All research described in this report was conducted on software and systems the author is authorised to examine — namely open-source code, the author's own systems, and the in-scope targets of vulnerability programmes whose rules the author follows. No unauthorised access to any computer, program or data was performed. Nothing in this report authorises, encourages, or should be construed as encouraging any act that would constitute an offence under the Computer Misuse Act 1990 or any other law. Vulnerabilities are disclosed privately and coordinated to a fix before any public detail, in accordance with the author's responsible-disclosure policy (Section 08).

6 · INTELLECTUAL PROPERTY AND TRADEMARKS

The design, layout and content of this report are © Stuart Thomas 2026, all rights reserved. The visual style is an original homage to the annual-report design of Paul Rand for IBM (c.1984); **no IBM trademark, logo, trade dress, image or other asset is reproduced**, and no association with, endorsement by, or origin from IBM or any other organisation is claimed or implied. "IBM" and all other third-party names appearing in this report are the trademarks of their respective owners and are used, where they appear at all, for identification and descriptive purposes only. The "TriageForge" name and striped device are the author's own.

7 · DATA PROTECTION

Personal data processed in the course of the research and of preparing this report has been handled in accordance with the UK General Data Protection Regulation and the Data Protection Act 2018. Where the work of third parties (for example, the maintainers who corrected findings) is described, it is described factually, from public records, and with respect; no personal data beyond names already public in project changelogs and correspondence is recorded here.

8 · LIMITATION OF LIABILITY

This report is provided for information only. To the fullest extent permitted by law, the author accepts no liability for any loss or damage arising from any use of, or reliance on, this report or its contents by any person. Nothing in these notices excludes or limits any liability that cannot lawfully be excluded or limited.

9 · STATEMENT ON AI ASSISTANCE

This report, and much of the research it describes, was prepared with the assistance of AI language models. In accordance with the author's methodology, every load-bearing factual claim was

intended to be verified against primary sources before inclusion, and material that could not be so verified was removed. The author uses AI assistance as a reasonable adjustment within the meaning of section 20 of the Equality Act 2010. Responsibility for the content of this report, and for every disclosure described in it, rests entirely with the author.

10 · GOVERNING LAW

These notices and this report are governed by, and construed in accordance with, the laws of England and Wales, and any dispute arising from them is subject to the exclusive jurisdiction of the courts of England and Wales.

Corporate information

The particulars, for the record.

THE PRACTICE

TriageForge
Trading name of Stuart Thomas, independent security researcher.
Whitby, North Yorkshire, United Kingdom.

PRINCIPAL

Stuart Thomas — Proprietor, Chairman & Principal Investigator.
Correspondence:
`stuart.thomas@triageforge.co.uk`

NATURE OF BUSINESS

Independent security research and coordinated vulnerability disclosure, with a specialism in memory-safety defects across hypervisors, operating-system infrastructure, and the parsers embedded in consumer platforms.

AUDITOR

None appointed. The accounts are reviewed annually by the proprietor, in the following October, in the course of writing the next of these reports — a process that has the twin merits of being free and of being conducted by the one person who knows where the numbers came from.

BANKERS

Retained, hopeful, and so far under-utilised for the purposes of this enterprise.

SELECTED PUBLICATION

The Calculator Discipline — A Taxonomy and Pre-Send Filter for AI-Assisted Vulnerability Disclosure Hallucinations. Stuart Thomas, 2026. Methodology paper and open-source tool; see Section 03.

THIS REPORT

TriageForge Annual Report 2026 — **public edition**. Prepared 31 October 2026. Issued in A4 print (PDF) and single-file HTML editions. Live and unremediated research is withheld; see the Legal & Regulatory Notices. Design: an original homage to Paul Rand. Photography: generated to brief. © Stuart Thomas 2026.

Prepared with AI assistance and verified against primary sources per the practice's own methodology. Every figure that could be checked, was. The one that mattered most was £0, and it checked out.



We found the bugs.
We told the people who can fix them.

TRIAGEFORGE · ANNUAL REPORT 2026 · PUBLIC EDITION

WHITBY · NORTH YORKSHIRE · UNITED KINGDOM

