Preprint, not peer reviewed

ThoxOS and ThoxInux: An Evidence-Bounded Architecture for a Local-First AI Operating System

The reviewed foundation (ThoxInux on Ginux) fixed as the architecture of record, the entitlement and update contracts tested as executable properties and fault simulations, the Linux-ABI ledger read at the pin and joined with a traced agent loop, and the kernel-model memory budget computed from the cards.

THOX.ai Research · Authors: Craig Ross, Chief Executive Officer (ORCID 0009-0005-9872-3673), and Tommy Xaypanya, CTO and Co-Founder, corresponding author (ORCID 0009-0000-4425-9565), THOX.ai LLC, Cedar Park, Texas

ABSTRACT

An operating system for local AI must carry an evidence chain from kernel behaviour to application execution, model quality and physical device operation, and the THOX record for ThoxOS™ has changed shape twice in four days: a 31 October whitepaper specified a Linux LTS fork, and a 1 November review draft identifies ThoxInux, a Ginux-derived Rust-first x86-64 kernel, as the reviewed foundation. This report fixes the architecture against that reviewed foundation and adds the evidence a general host can produce. We implement the entitlement contract clean-room in two token designs and show by executable properties that the shared-secret HMAC design accepts a token forged by any device that can verify one, while an Ed25519 design rejects it at 196 microseconds per verification on a 2-vCPU host, and that a wall-clock grace window is revived by a clock rollback unless a persisted high-water mark is used. We simulate the A/B update contract under injected faults: the full contract ends every one of 200 updates on a known-good slot with at most 3 boots of recovery, while removing the signature check boots 6 tampered images and removing the attempt counter or the second slot leaves about 26 devices unbootable per run. We read the upstream Linux-ABI ledger at the exact commit ThoxInux pins (375 dispatched syscalls: 82 Complete, 245 Partial, 48 Policy-denied) and join it with a strace of a reference agent loop on the authoring host: of 56 distinct syscalls the loop issues, 10 are Complete, 46 Partial and 0 Policy-denied, so a plain agent loop touches nothing the policy refuses and almost nothing that is proven equivalent to Linux. We compute the memory budget for the three published kernel models from their card geometry: the 0.8B student fits the 2 GB Air floor with 36,769 tokens of context at a 1 GB reserve and nothing at 1.5 GB; the 4B student fits the 4 GB Mini floor only at 7,350 tokens; the 27B teacher needs a host above 16 GB. No ThoxInux boot, no device, no model inference and no production security were executed here; the validation protocol that would produce those results is restated with its gates.

CONTENTS

Executive summary
1. Introduction
2. Background and related work
3. System and threat model
4. Design
5. Implementation
6. Evaluation
7. Discussion and limitations
8. Conclusion
9. Roadmap and decisions for the founders
Appendix A. Reproducibility
Appendix B. Notation

Executive summary

ThoxOS™ is the operating system for THOX's local-first AI devices and desktops. Its record has two foundations in four days: the v0.1.0 whitepaper of 31 October specified linux-thox, a Linux LTS fork with a separate Rust research laboratory [1]; the 0.2.0-review draft of 1 November identifies ThoxInux, a Ginux-derived Rust-first x86-64 kernel with Linux userspace compatibility, as the reviewed foundation, and treats the whitepaper as historical input [2, 3]. This report takes the reviewed foundation as the architecture of record, keeps what the whitepaper specified that survives the change (class floors, entitlement without DRM, A/B updates with a health gate, the open and paid line), and adds the evidence a general host can produce.

196 µs
Ed25519 token verify on a 2-vCPU host; HMAC 12 µs. HMAC accepts a verifier-forged token, Ed25519 rejects it
MEASURED

200 / 200
updates ending on a known-good slot under the full A/B contract with injected faults, at most 3 recovery boots; no signature check boots 6 tampered images
SIMULATED

10 / 56
distinct syscalls of a traced reference agent loop that are Complete at the ThoxInux pin; 46 Partial, 0 Policy-denied
OBSERVED + MEASURED

7,350 tok
largest context for ThoxOS-Kernel-4B Q4_K_M resident on the 4 GB Mini floor at a 1 GB reserve; no fit at 1.5 GB
CALCULATED

PRODUCT IMPLICATIONS

First, the entitlement reference implementation's HMAC demo cannot be the production design: any device holding the verification secret can mint tokens, and the fix is a public-key signature plus a persisted time high-water mark, both of which this report tests. Second, the A/B contract's value is in the attempt counter and the second slot, not in the health gate alone; dropping either leaves about 26 of 200 updates unbootable in simulation. Third, the Linux ABI at the pin is dispatched but mostly Partial, and a plain agent loop lives almost entirely in Partial calls, so P1 of the review protocol (resident-agent lifecycle on the real payload) is the first measurement to fund. Fourth, the 4B student fits the Mini floor only at short context; the Mini's resident model is a context decision, and the product copy should say so.

1 Introduction

Local-first software keeps the user's data and computation on hardware the user owns [4]. An operating system built for local AI inherits that promise and adds four

layers that each need their own evidence: the kernel must sustain the workload's process, memory, filesystem and networking behaviour; the runtime must load the exact model artifact; the application must enforce the user's authority; and the system must fit the hardware envelope. The 0.2.0 review states this chain and the rule that evidence at one layer never establishes another [2]. This report follows that rule and applies it to the foundation change.

ThoxInux is a derivative of Gregory M. Kurtzer's Ginux, an independent Rust-first x86-64 kernel with Linux userspace compatibility, a ring-3 administration controller and a UEFI boot image [5]. ThoxInux pins upstream commit 36503bda and ships engineering prereleases v0.9.0 and v0.9.1 whose unified EFI payload passed a 22-check lifecycle suite under QEMU TCG and KVM on one host; its ledger also records a resident agent failure on an earlier edition payload and conditional build reproducibility [3]. Those are repository-reported runs, not results of this report. The whitepaper's Linux LTS fork position is retained as a dated design whose entitlement, class-floor and update specifications carry over unchanged [1].

The contributions are:

1. **An architecture of record on the reviewed foundation** (Section 4) with the whitepaper's surviving contracts restated as numbered requirements and the review's two equations made executable.
2. **Entitlement evidence** (Section 6.1): a clean-room implementation of the Core-never-gated invariant and the MayExecute conjunction in two token designs, with ten executable properties; the shared-secret design accepts a verifier-forged token and the public-key design rejects it, at 196 microseconds per verification; a wall-clock grace window is revived by a clock rollback unless a persisted high-water mark is kept.
3. **Update evidence by simulation** (Section 6.2): the full A/B contract ends all 200 updates on a known-good slot under injected faults with at most 3 boots of recovery; ablating the signature check, the attempt counter or the second slot each produces a named failure class.
4. **Linux ABI evidence** (Section 6.3): the upstream ledger at the pin classified by area, and a strace of a reference agent loop joined with the ledger: 10 of 56 distinct calls Complete, 46 Partial, none Policy-denied.
5. **Memory budget evidence** (Section 6.4): the review's equation 3 evaluated for the three published kernel models from their card geometry against the class floors, including the losing regimes.

Nothing in this report booted ThoxInux, ran a model, touched a device or evaluated production security. Section 7 says so itemised; Section 9 lists the decisions only the founders can take.

2 Background and related work

2.1 Rust kernels with Linux ABI compatibility

Asterinas is a Rust framekernel with Linux ABI compatibility and a small trusted computing base, evaluated at USENIX ATC 2025 [6, 7]; Rust-for-Linux brings Rust into the Linux kernel itself [8]. Ginux takes a third position: an independent kernel that dispatches the pinned Linux v6.12 ×86-64 syscall table and ledgers each call as Complete, Partial, Policy-denied or Missing, with Linux's documented stable ABI as the authority [5, 9]. Neither Asterinas's safety argument nor its measurements transfer to Ginux or ThoxInux; the review says this and this report repeats it.

2.2 Agent operating systems and privilege

A line of work frames the LLM as an operating-system component or the OS as an agent host: AIOS and its vision paper [10, 11], a lessons-from-classical-OS treatment [12], and an agent framework for Linux schedulers [13]. ThoxOS's position is the opposite of LLM-as-OS: the model proposes and the harness disposes, and the review's MayExecute conjunction makes the model a non-term. Progent's privilege control for agents [14] and the LITMUS and SafeClawBench evaluations of agent behaviour in real OS environments and sandboxes [15, 16] are the closest evaluation precedents for the review's P2 protocol.

2.3 Updates, entitlement and model reporting

RAUC documents the A/B slot, bootloader and mark-good integration this design adopts [17]. HMAC and EdDSA are the two primitives the entitlement experiment compares [18, 19]. Model Cards motivate treating the ThoxOS-Kernel cards as release documentation whose conditions travel with the numbers [20]; the cards themselves are the source of the geometry and file sizes used here [21, 22, 23].

Source	Kernel position	Entitlement	Updates	Status of evidence
ThoxOS whitepaper v0.1.0 [1]	Linux LTS fork (linux-thox), Rust lab separate	signed offline token, 30-day grace, Core never gated	RAUC A/B, health gate	staging tests reported (25 entitlement, 27 image checks); no device
0.2.0-review draft [2]	ThoxInux (Ginux-derived, Rust-first x86-64)	eq. 1 and eq. 2; HMAC demo flagged as issuer-trust failure	RAUC template; P4 protocol	repository-reported runs; protocol unexecuted
ThoxInux repository [3]	Ginux pin 36503bda; v0.9.0 and v0.9.1 prereleases	n/a	n/a	22/22 lifecycle under TCG and KVM on one host; resident-agent fault on an edition payload

Source	Kernel position	Entitlement	Updates	Status of evidence
This report	ThoxInux as architecture of record	two designs tested; forgery and rollback hazards shown	contract simulated with ablations	measured on a host, observed ledger, simulated protocol, calculated budget

Table 1. The THOX record and this report. Evidence classes are named per row; nothing in the first three rows is re-run here.

3 System and threat model

3.1 Setting

Devices are the E2, P4 and P6 classes (ThoxAir 2 GB, ThoxMini 4 GB, ThoxClip 6 GB RAM floors) plus host-tier machines; the ThoxKey companion runs no resident model and is served by a connected host [1, 21]. The reviewed kernel target is x86-64; Arm is scoped, not ported [2]. MeshStack™ is the optional private mesh and ThoxWork™ the workspace; their roles are integration boundaries, not evidence that the stack has executed on the reviewed kernel [24].

3.2 Requirements

1. **R1 Core never gated.** $f \in F_{\text{core}} \Rightarrow E(f, t) = \text{true}$ for every token state t (review eq. 1).
2. **R2 Authorization is a conjunction.** $\text{MayExecute}(a) = U(a) \wedge O(a) \wedge E(f(a), t)$; model output sets none of the terms (review eq. 2). Unknown feature identifiers are denied explicitly.
3. **R3 Issuer-only trust.** A device that can verify an entitlement cannot issue one.
4. **R4 Clock discipline.** A rolled-back clock cannot revive an expired entitlement.
5. **R5 Signed bundles, A/B slots, bounded attempts, health gate, data untouched.** Any failure returns the device to the last known-good slot without user action.
6. **R6 Memory budget.** $M_{\text{weights}} + M_{\text{KV}} + M_{\text{runtime}} + M_{\text{OS}} + M_{\text{reserve}} \leq M_{\text{available}}$ (review eq. 3), evaluated per model, quantisation, class and context.
7. **R7 ABI truth.** Compatibility claims cite the ledger status at the pin, never a count of dispatched calls.
8. **R8 Evidence separation.** Observed artifacts, repository-reported execution, publisher-reported evaluation and proposed protocol are never merged (review Section 3.1).

3.3 Threat model

In scope: untrusted prompts and tool input, a compromised application, a malformed or tampered update bundle or entitlement token, a device that holds

verification material, and a device whose clock is set back. Out of scope: a compromised issuer, a compromised kernel, physical key extraction, and side channels. The trusted boundary is the authenticated user decision, the enforcing broker, OS access control and verified artifact identity; the model is never in it [2].

4 Design

4.1 Architecture of record

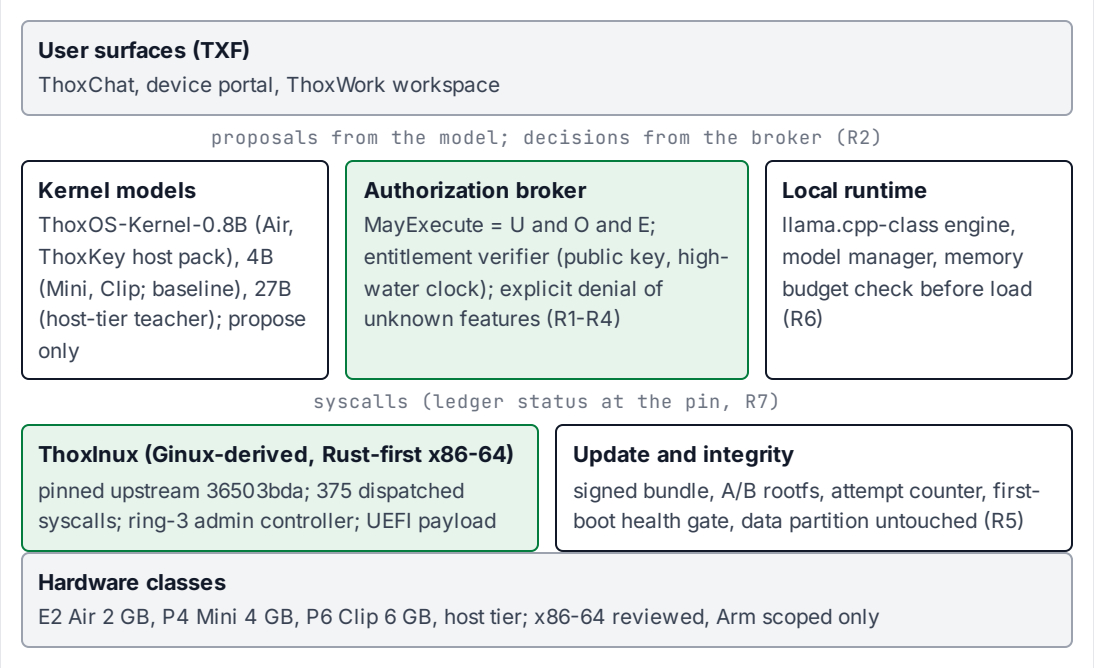


Figure 0. Architecture of record on the reviewed foundation. Schematic; the model layer proposes, the broker decides, the kernel is ThoxInux at the pin, and every box carries the evidence class of Section 6.

4.2 Entitlement without DRM, made executable

A token is a payload (account, feature set, issued-at, expiry, nonce) and a signature. Validity is signature correctness, a well-formed payload, membership of the requested feature, and $\mathrm{now} \le \exp + G$ with G the 30-day grace. Core features short-circuit before any token is examined, which is how R1 is enforced rather than hoped. The whitepaper's reference implementation and the review's demo use HMAC under a shared secret [1, 2]; the review names the consequence, that a holder of the verification secret can also generate MACs. This report tests two designs side by side:

$$\mathrm{hmac}:\ \tau = p \,\|\, \mathrm{HMAC}_k(p), \quad \text{verifier holds } k; \qquad \mathrm{ed25519}:\ \tau = p \,\|\, \mathrm{Sig}_{sk}(p), \quad \text{v}$$

Proposition 1 (forgeability). Under the hmac design any party holding k produces a token the verifier accepts for any feature set; under the ed25519 design a party holding only pk produces an accepted token with probability bounded by EdDSA's existential unforgeability. *Proof.* HMAC verification recomputes $\mathrm{HMAC}_k(p)$, so possession of k is possession of the issuing capability; for Ed25519 the statement is the EUF-CMA property of the scheme [19]. □ Property P9 in Section 6.1 is the executable form.

Proposition 2 (clock rollback). With wall-clock time only, an expired token with expiry e is accepted again whenever the clock is set to any $t' \le e + G$; with the device persisting $h = \max(h, \mathrm{wall})$ and evaluating validity at h, a rollback changes nothing. *Proof.* h is monotone by construction and already exceeded $e + G$ when the token expired. □ Property P10 is the executable form. Persisting h needs no trusted time source; it needs tamper-evident local storage, which the review lists as production work.

4.3 The update contract

Algorithm 1. First boot of a new slot (health gate with bounded attempts)

```
Precondition: bundle signature and image hashes verified before any write; image written
to the idle slot;
              attempt counter armed with N = 3; data partition untouched
boots ← 0
loop:
    boots ← boots + 1; boot idle slot
    if thoxd answers, portal is enabled, local mesh discovery is up, network reaches
targets, slot reports bootable:
        mark slot GOOD; it becomes the running slot; return OK
    if boots ⩾ N: bootloader returns to the previous GOOD slot; mark idle slot BAD;
return ROLLED_BACK
```

Three ablations are simulated in Section 6.2: no signature check (R5 first clause), no attempt counter, and a single slot updated in place. Each removes one clause of the contract and each produces a distinct failure class, which is the argument that the contract has no redundant part.

4.4 ABI truth and the agent path

Ginux's ledger scores every dispatched syscall; ThoxInux's audit recounts it and runs upstream's architecture checker at the pin [3]. This report adds an area classification and a measured join: which calls a plain agent loop issues and what their status is. The loop (Appendix A) reads a configuration, appends a log, spawns a child and waits, opens a TCP connection to a local listener, polls, handles a signal and maps a file. It is a Python program on the authoring host's Linux; it is not the ThoxOS agent and it did not run on ThoxInux. Its value is to name the Partial calls that P1 of the review protocol must exercise first.

4.5 Memory budget

For a Qwen3.5-family hybrid model with n_f full-attention layers of n_{kv} KV heads and head dimension d, bf16 KV costs $2\,n_f\,n_{kv}\,d \cdot 2$ bytes per token; the linear-attention layers carry a fixed fp32 state. From the cards: 0.8B 12,288 B/token, 4B 32,768 B/token, 27B 65,536 B/token [21, 22, 23]. Weights are the published GGUF sizes, assumed fully resident. The reserve term is swept because it is not measured.

5 Implementation

Three implementations exist in the record and one in this report. ThoxInux carries the kernel, a Python lifecycle suite, source-preparation and launcher tests, an SSD installer with tests and an agentos Rust crate [3]; the historical thoxos-core lane carries the entitlement reference (25 tests) and the image validator (27 checks) [1]. This report could not execute code from those public clones under the authoring session's policy, so their test states are quoted as repository-reported. This report's own code is Appendix A: the entitlement contract in two designs with property tests, the update simulator, the ledger parser and traced loop, and the memory budget calculator.

Requirement	Specified	Test or evidence in this report	Result
R1 Core never gated	Section 4.2	P6: Core allowed with no, expired, tampered and garbage tokens	pass
R2 conjunction, unknown denied	Section 4.2	P7, P8	pass
R3 issuer-only trust	Prop. 1	P9: forged token accepted (hmac) / rejected (ed25519)	accepted / rejected
R4 clock discipline	Prop. 2	P10: rollback with wall clock / with high-water mark	accepted / rejected
R5 update contract	Algorithm 1	exp/update_sim.py, 30 seeds x 200 updates, 4 policies	simulated; Section 6.2
R6 memory budget	Section 4.5	exp/memory_budget.py over cards, floors, contexts, reserves	calculated; Section 6.4
R7 ABI truth	Section 4.4	exp/syscall_ledger.py: ledger parse (375 rows asserted) and strace join	observed and measured; Section 6.3
R8 evidence separation	throughout	every number carries its class in summary.json	editorial

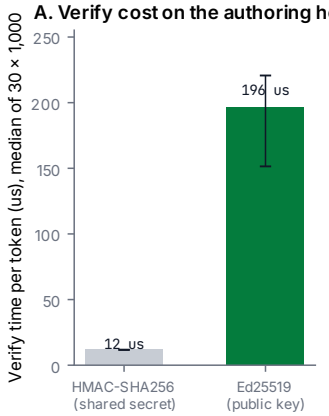
Table 2. Requirement-to-evidence map.

6 Evaluation

Setup. Authoring container, Linux, 2 vCPU, 7 GB RAM, Python 3.13, cryptography library for Ed25519, strace for the traced loop. Timing: 30 repetitions of 1,000 verifications after warm-up, medians with interquartile ranges. The ledger was fetched from the upstream repository at the pinned commit on 2026-11-03. Model geometry and sizes were read from the three public cards the same day. Nothing ran on ThoxInux.

6.1 Does the entitlement contract hold, and which token design is safe?

A. Verify cost on the authoring host



B. Property outcomes (red = the hazard the design leaves open)

	HMAC design	Ed25519 design
P1 valid paid	pass	pass
P2 grace +29 d	pass	pass
P3 grace +31 d	pass	pass
P4 tampered	pass	pass
P5 malformed	pass	pass
P6 Core unconditional	pass	pass
P7 U and O required	pass	pass
P8 unknown feature	pass	pass
P9 forged by verifier holder	ACCEPTED	rejected
P10 clock rollback (wall)	ACCEPTED	ACCEPTED
P10 clock rollback (hwm)	pass	pass

Figure 1. Entitlement contract in two token designs. A: verify cost per token on the authoring host (median of 30 × 1,000). B: the ten properties; P9 and P10 are hazards, so a red ACCEPTED is the finding. Token sizes: 198 bytes (hmac) and 241 bytes (ed25519).

Properties P1 to P8 pass in both designs: Core is unconditional, the conjunction holds, grace admits at 29 days and refuses at 31, tampered and malformed tokens are refused, unknown features are denied. P9 separates the designs: a forged token minted with the verification secret is accepted under hmac and rejected under ed25519, which is Proposition 1 run. P10 is design-independent: with wall-clock validity a token expired 40 days earlier is accepted after a 20-day clock rollback, and with a persisted high-water mark it is rejected. The cost of the safe design is 196 microseconds per verification against 12 for HMAC on this host, a difference that no entitlement check on a device will notice. The recommendation is unambiguous: production tokens are Ed25519-signed, devices hold only the public key, and the device persists its time high-water mark.

6.2 Does the A/B contract recover, and which clause does the work?

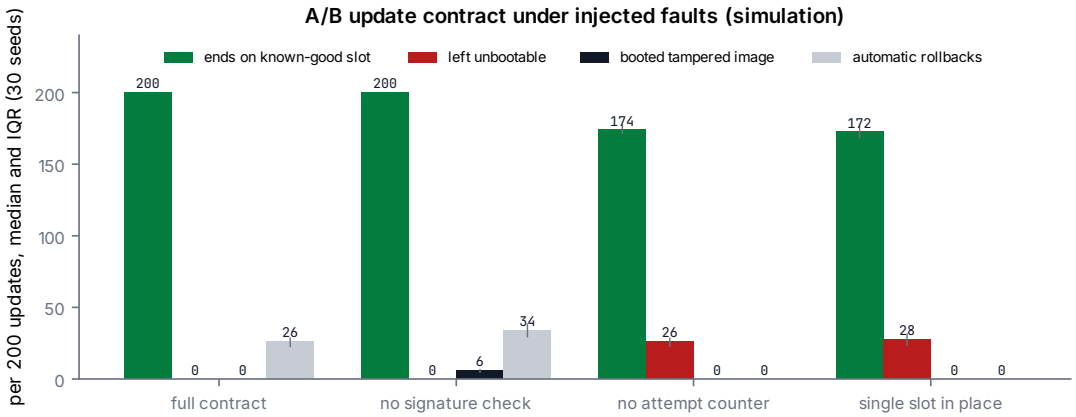


Figure 2. A/B update contract under injected faults, 30 seeds of 200 updates each, medians with IQR. Fault probabilities per update: bad signature 3%, incomplete write 5%, corrupt image 3%, health failure 6%, power loss in the health window 4%, boot hang 4%. Protocol logic only.

The full contract ends all 200 updates on a known-good slot, with 26 automatic rollbacks and never more than 3 boots of recovery. Removing the signature check keeps the device bootable but boots 6 tampered images per run; removing the attempt counter leaves 26 devices looping on a bad slot; a single slot updated in place ends 172 updates well and leaves 28 unbootable. The three clauses fail differently, so none is redundant, and the ablation ordering tells the release team what to test first on hardware: attempt-counter exhaustion and interrupted writes (P4 of the review protocol), not the happy path.

6.3 What does the Linux ABI at the pin look like, and what does an agent loop touch?

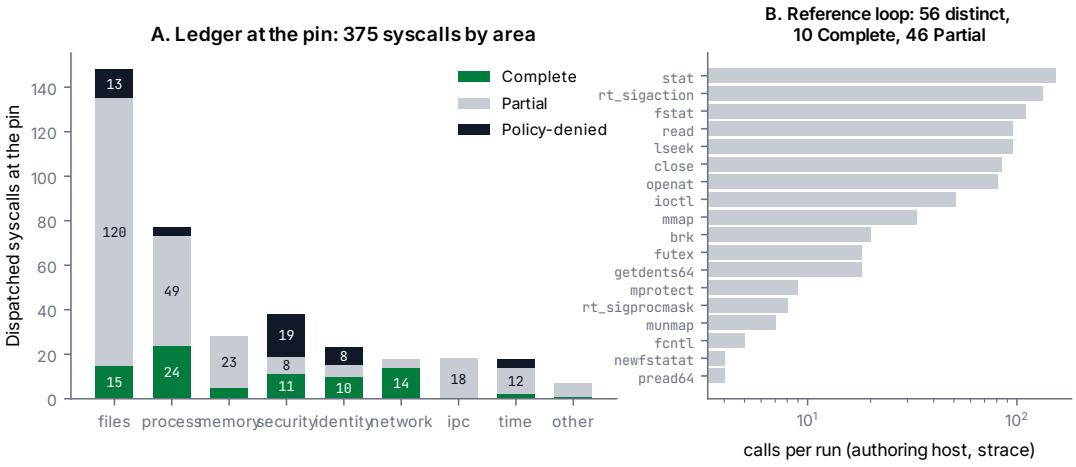


Figure 3. A: the upstream ledger at commit 36503bda, 375 dispatched syscalls, classified by area (classification is this report's; statuses are the ledger's). B: the 18 most frequent syscalls of the reference agent loop under strace on the authoring host, coloured by ledger status; none is Policy-denied.

At the pin, 82 syscalls are Complete, 245 Partial and 48 Policy-denied, with nothing Missing [5, 3]. By area, files is 120 Partial to 15 Complete, network is 14 Complete to 4 Partial, and the Policy-denied set is concentrated in security (19) and identity (8): credential mutation, bpf and Landlock are refused by design, which matches the ThoxInux domain model. The reference loop issues 56 distinct syscalls and 991 calls per run: 10 distinct calls are Complete, 46 Partial, 0 Policy-denied and 0 absent from the pinned table; by volume 975 of 991 calls land in Partial entries (stat, rt_sigaction, read, openat, mmap, futex, clone3, wait4 and their neighbours). The positive reading is that an agent loop asks for nothing the policy refuses. The cautionary reading is that it lives almost entirely in calls that are dispatched and useful but not proven equivalent to Linux, which is exactly the surface on which the edition payload's resident-agent failure occurred [2]. This is the evidence for funding P1 before any other protocol item.

6.4 What fits where? (Calculated)

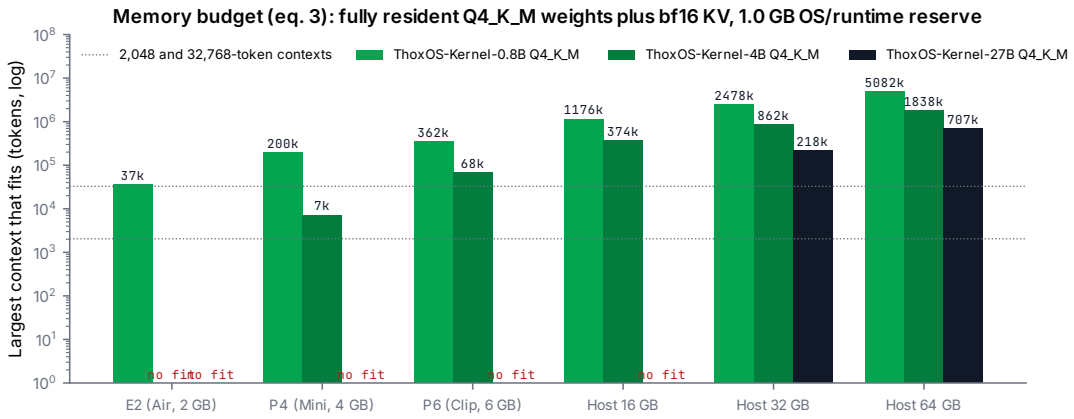


Figure 4. Largest context that fits the review's memory budget for each published kernel model at Q4_K_M with fully resident weights, bf16 KV from the card geometry, fixed linear-attention state, and a 1.0 GB OS-plus-runtime reserve. 'No fit' means the weights and reserve alone exceed the floor. Calculated; nothing measured.

The 0.8B student fits the 2 GB Air floor with 36,769 tokens at a 1.0 GB reserve and no fit at 1.5 GB: the reserve, not the model, decides the Air. The 4B student fits the 4 GB Mini floor only at 7,350 tokens at a 1.0 GB reserve, no fit at 1.5 GB, and its Q8_0 build no fit; on the 6 GB Clip it reaches 68,385 tokens. The 27B teacher is no fit on a 16 GB host at Q4_K_M, 218,225 tokens on 32 GB, and at Q8_0 34,381 tokens on 32 GB and 522,662 tokens on 64 GB. Two product consequences: the Mini's resident 4B model is a 2k-to-8k-context assistant unless the reserve is measured smaller than 1 GB, and the 27B host tier starts at 32 GB. File size is a lower bound on resident weight memory and the reserve is an assumption; the review's warning that storage bytes, advertised memory and peak resident memory are different quantities stands [2].

7 Discussion and limitations

1. **No kernel execution.** ThoxInux was not booted here; its 22/22 lifecycle results, the resident-agent fault and the reproducibility conditions are repository-reported [3].
2. **Policy-bound.** The authoring session could not run code from the public ThoxInux clone, so its suites were not re-run; this report's own code ran.
3. **The traced loop is not the ThoxOS agent.** It is a Python program on the authoring host's Linux; it establishes which ledger entries a plain loop touches, not that the loop works on ThoxInux.
4. **Area classification is ours.** The ledger's statuses are upstream's; the nine areas are this report's regex classification of syscall names.
5. **Entitlement scope.** The two designs are clean-room; production key custody, rotation, revocation, replay and trusted time remain unbuilt, as the review lists.
6. **Update simulation.** Fault probabilities are stated assumptions; the simulation evaluates contract logic, not RAUC, a bootloader or flash behaviour.
7. **Memory budget.** GGUF size is a lower bound on resident weights; the reserve is swept, not measured; the linear-attention state is an estimate from config.json.
8. **Record motion.** The whitepaper (31 October) states no kernel-model weights exist; the cards (1 November) publish them, and the thox.ai paper page reads '27B card only, weights not public' while the 27B card lists public GGUF and safetensors [25, 23]. This report follows the cards and flags the page for correction.
9. **No device, no model run, no security assessment.** No E2, P4 or P6 hardware was touched; no inference ran; no penetration or adversarial test was performed.
10. **Not peer reviewed.** Preprint prepared by THOX.ai Research with AI assistance (Appendix C).

8 Conclusion

ThoxOS's foundation changed from a Linux fork to ThoxInux in the week before this report, and the contracts the whitepaper specified survived the change because they never depended on the kernel. This report made two of them executable and tested them: the entitlement contract, whose demo design is forgeable by any verifier and whose grace window is revivable by a clock rollback, both fixed by a public key and a persisted high-water mark; and the update contract, whose every clause removes a distinct failure class in simulation. It read the ABI ledger at the pin and showed that a plain agent loop asks for nothing refused and almost nothing proven. It computed what fits on each device from the

cards. The next evidence is the review's P1 on the real payload bytes, and this report says why it comes first.

9 Roadmap and decisions for the founders

1. **Record correction.** Align the thox.ai paper page and the whitepaper status box with the 1 November cards (27B weights public).
2. **Entitlement production design.** Adopt Ed25519 tokens with device-held public keys and a persisted time high-water mark; the HMAC demo stays a demo.
3. **P1 first.** Run the review's resident-agent lifecycle protocol on Payload U and v0.9.1 bytes, 30 cold starts per configuration, with the Partial calls of Section 6.3 as the watch list.
4. **P4 on hardware.** Interrupted writes and attempt-counter exhaustion before the happy path.
5. **Measure the reserve.** One measured OS-plus-runtime figure per class replaces the sweep and settles the Mini context statement.
6. **Venue.** The roster is settled (Craig Ross and Tommy Xaypanya, corresponding author Tommy Xaypanya); the remaining venue decision is whether this edition or the P1-measured successor is the one submitted.

A Appendix A. Reproducibility

Commands (from the artifact root)

```
pip install cryptography numpy matplotlib pypdf pillow && cd wp && npm install && cd ..
mkdir -p ginux_ledger && curl -o ginux_ledger/LINUX-COMPATIBILITY.md
https://raw.githubusercontent.com/gmkurtzer/ginux/36503bda2d87465765e1b94e65f9d100f27ed5be
COMPATIBILITY.md
python3 exp/entitlement_experiments.py   # fails on any property violation
python3 exp/update_sim.py
python3 exp/memory_budget.py
python3 exp/syscall_ledger.py            # needs strace; asserts 375 ledger rows
python3 exp/analyze.py && python3 wp/figures.py && python3 wp/build.py
node wp/katex_render.cjs wp/paper.src.html wp/paper.html && node wp/pdf.cjs
wp/paper.html wp/cover.pdf wp/body.pdf && python3 wp/merge_pdf.py
```

Entitlement known-answer vector (hmac design, secret = bytes 0x00..0x1f): payload `{"acct":"acct-demo","exp":1831536000,"feat":["meshstack-pro"],"iat":1800000000,"nonce":"QkJCQkJCQkJCQkJCQkJCQg","v":1}`; token `eyJhY2N0IjoiYWNjdC1kZW1vIiwiZXhwIjoxODMxNTM2MDAwLCJmZWF0IjpbIm1lc2hzdGFjay1wcm8iXSwiaWF0IjoxODAwMDAwMDAwLCJub25jZSI6IlFrSkNRa0pDUWtKQ1FrSkNRa0pDUWciLCJ2IjoxfQ.N3P1z7mcN6DvXBGQm0o9WOtosb4QAhqBGuFn8pYf9IE`; allowed at 1800086400, allowed in grace at 1834041600, denied at 1834214400 (Unix seconds).

Reference loop last log line on the authoring host: `{"data": "hello", "child": "child", "sig": 1, "model": "ThoxOS-Kernel-0.8B"}`.

B Appendix B. Notation

Symbol	Meaning
$F_{\text{core}}, E(f,t)$	Core feature set; entitlement decision for feature f under token state t
$U(a), O(a), f(a)$	user authorization, OS and application access control, feature of action a
G, h	grace window (30 days); persisted time high-water mark
k, sk, pk	HMAC secret; Ed25519 private and public key
n_f, n_{kv}, d	full-attention layers, KV heads, head dimension
E2, P4, P6	THOX compute classes (Air, Mini, Clip)

C Appendix C. Authorship and disclosure

Issued by THOX.ai Research. The authors are Craig Ross (Chief Executive Officer) and Tommy Xaypanya (Chief Technology Officer and Co-Founder), THOX.ai LLC; Tommy Xaypanya is the corresponding and accountable human author. Author consent to this edition is recorded. Generative AI assistance: experiment code, analysis, figures and draft text were produced by an AI system (Claude, Anthropic) operating as the THOX digital human Odette Lark under the direction of the corresponding author; every citation was resolved live on 2026-11-03 and every number is generated from the experiment outputs. No AI system is an author. The authors are THOX.ai principals; this is an interested-party technical account. Upstream attribution: ThoxInux derives from Gregory M. Kurtzer's Ginux under its recorded terms. Edition v1.0.0 was an internal draft; this edition, v1.0.1, is the public preprint edition authorised by the founders: it settles the author roster, retires the internal marking and changes no result, measurement or conclusion. The 0.2.0-review draft it builds on is public on Zenodo.

References

[1] THOX.ai. *ThoxOS: An Open Kernel and Operating System for Local-First AI (technical whitepaper v0.1.0)*. THOX.ai, 2026-10-31; historical design under CC BY 4.0 (Linux LTS fork position, class floors, entitlement and A/B design).

[2] Craig Ross, Tommy Xaypanya. *ThoxOS and ThoxInux: Architecture, Evidence, and a Validation Protocol for Local-First AI (technical research draft, revision 0.2.0-review)*. THOX.ai LLC, 2026-11-01; version of record on Zenodo, DOI 10.5281/zenodo.23876230. https://doi.org/10.5281/zenodo.23876230

[3] THOX.ai. *ThoxInux: source repository, compatibility audit (docs/COMPATIBILITY.md), wave ledger (docs/WAVES.md) and release v0.9.0 (Payload U SHA-256 85722bd4…4420ae)*. GitHub, main at

334539c3 cloned 2026-11-03; v0.9.0 engineering prerelease. https://github.com/Thox-ai/thoxinux

[4] Martin Kleppmann, Adam Wiggins, Peter van Hardenberg, Mark McGranaghan. *Local-first software: you own your data, in spite of the cloud*. Onward! 2019; Ink and Switch essay. https://www.inkandswitch.com/local-first/

[5] Gregory M. Kurtzer and Ginux contributors. *Ginux: independent kernel and Linux userspace compatibility (repository; LINUX-COMPATIBILITY.md at commit 36503bda2d87465765e1b94e65f9d100f27ed5be)*. GitHub, fetched 2026-11-03. https://github.com/gmkurtzer/ginux

[6] Yuke Peng, Hongliang Tian, Junyang Zhang, Ruihan Li, Chengjun Chen, Jianfeng Jiang, Jinyi Xian, Xiaolin Wang, Chenren Xu, Diyu Zhou, Yingwei Luo, Shoumeng Yan, Yinqian Zhang. *ASTERINAS: A Linux ABI-compatible, Rust-based framekernel OS with a small and sound TCB*. USENIX ATC 2025, pp. 307-323. https://www.usenix.org/conference/atc25/presentation/peng-yuke

[7] Yuke Peng, Hongliang Tian, Zhang Junyang, Ruihan Li, Chengjun Chen, Jianfeng Jiang, Jinyi Xian, Xiaolin Wang, Chenren Xu, Diyu Zhou, Yingwei Luo, Shoumeng Yan, Yinqian Zhang. *Asterinas: A Linux ABI-Compatible, Rust-Based Framekernel OS with a Small and Sound TCB*. arXiv:2506.03876v1, first posted 2025-06-04, version inspected 2025-06-04; USENIX ATC 2025. https://arxiv.org/abs/2506.03876

[8] Shane K. Panter, Nasir U. Eisty. *Rusty Linux: Advances in Rust for Linux Kernel Development*. arXiv:2407.18431v2, first posted 2024-07-25, version inspected 2024-09-06. https://arxiv.org/abs/2407.18431

[9] The Linux kernel developers. *Linux ABI stability (admin-guide/abi-stable)*. kernel.org documentation, accessed 2026-11-03. https://www.kernel.org/doc/html/latest/admin-guide/abi-stable.html

[10] Kai Mei, Xi Zhu, Wujiang Xu, Wenyue Hua, Mingyu Jin, Zelong Li, Shuyuan Xu, Ruosong Ye, Yingqiang Ge, Yongfeng Zhang. *AIOS: LLM Agent Operating System*. arXiv:2403.16971v5, first posted 2024-03-25, version inspected 2025-08-12. https://arxiv.org/abs/2403.16971

[11] Yingqiang Ge, Yujie Ren, Wenyue Hua, Shuyuan Xu, Juntao Tan, Yongfeng Zhang. *LLM as OS, Agents as Apps: Envisioning AIOS, Agents and the AIOS-Agent Ecosystem*. arXiv:2312.03815v2, first posted 2023-12-06, version inspected 2023-12-09. https://arxiv.org/abs/2312.03815

[12] Gosia Steinder, Hubertus Franke. *Towards an Agent Operating System - Lessons from Classical and Cloud OS*. arXiv:2607.25076v1, first posted 2026-07-27, version inspected 2026-07-27. https://arxiv.org/abs/2607.25076

[13] Yusheng Zheng, Yanpeng Hu, Wei Zhang, Andi Quinn. *Towards Agentic OS: An LLM Agent Framework for Linux Schedulers*. arXiv:2509.01245v4, first posted 2025-09-01, version inspected 2025-09-30; MLforSystem 2025. https://arxiv.org/abs/2509.01245

[14] Tianneng Shi, Jingxuan He, Zhun Wang, Hongwei Li, Linyu Wu, Wenbo Guo, Dawn Song. *Progent: Securing AI Agents with Privilege Control*. arXiv:2504.11703v3, first posted 2025-04-16, version inspected 2026-05-14. https://arxiv.org/abs/2504.11703

[15] Chiyu Zhang, Huiqin Yang, Bendong Jiang, Xiaolei Zhang, Yiran Zhao, Ruyi Chen, Lu Zhou, Xiaogang Xu, Jiafei Wu, Liming Fang, Zhe Liu. *LITMUS: Benchmarking Behavioral Jailbreaks of LLM Agents in Real OS Environments*. arXiv:2605.10779v1, first posted 2026-05-11, version inspected 2026-05-11. https://arxiv.org/abs/2605.10779

[16] Yuchuan Tian, Mengyu Zheng, Haocheng Mei, Ye Yuan, Chao Xu, Xinghao Chen, Hanting Chen, Yu Wang. *SafeClawBench: Separating Semantic, Audit-Evidence, and Sandbox Harm in Tool-Using LLM Agents*. arXiv:2606.18356v1, first posted 2026-06-16, version inspected 2026-06-16. https://arxiv.org/abs/2606.18356

[17] RAUC developers. *RAUC integration documentation: bootloader integration, system health, mark-good*. rauc.readthedocs.io, accessed 2026-11-03. https://rauc.readthedocs.io/en/latest/integration.html

[18] Hugo Krawczyk, Mihir Bellare, Ran Canetti. *HMAC: Keyed-Hashing for Message Authentication*. RFC 2104, IETF, 1997. https://www.rfc-editor.org/rfc/rfc2104

[19] Simon Josefsson, Ilari Liusvaara. *Edwards-Curve Digital Signature Algorithm (EdDSA)*. RFC 8032, IETF, 2017. https://www.rfc-editor.org/rfc/rfc8032

[20] Margaret Mitchell, Simone Wu, Andrew Zaldivar, Parker Barnes, Lucy Vasserman, Ben Hutchinson, Elena Spitzer, Inioluwa Deborah Raji, Timnit Gebru. *Model Cards for Model Reporting*. arXiv:1810.03993v2, first posted 2018-10-05, version inspected 2019-01-14; FAT* '19: Conference on Fairness, Accountability, and Transparency, January 29--31, 2019, Atlanta, GA, USA. https://arxiv.org/abs/1810.03993

[21] THOX.ai. *ThoxOS-Kernel-0.8B model card and repository (config.json, GGUF sizes)*. Hugging Face, read 2026-11-03. https://huggingface.co/Thox-ai/ThoxOS-Kernel-0.8B

[22] THOX.ai. *ThoxOS-Kernel-4B model card and repository*. Hugging Face, read 2026-11-03. https://huggingface.co/Thox-ai/ThoxOS-Kernel-4B

[23] THOX.ai. *ThoxOS-Kernel-27B model card and repository*. Hugging Face, read 2026-11-03. https://huggingface.co/Thox-ai/ThoxOS-Kernel-27B

[24] THOX.ai Research. *THOX MeshStack: Capacity-Priced Pipeline Sharding, Grant-Gated KV Transfer and Lease-Fenced Device Roles for a Private Household Mesh (technical report v1.0.0)*. THOX.ai Research, 2026-11-03, preprint; ttracx/thox-meshstack-inference papers/research-paper-v1.0.0.

[25] THOX.ai. *ThoxOS and ThoxInux, THOX Research Paper (web page for revision 0.2.0-review)*. thox.ai, read 2026-11-03. https://www.thox.ai/research/papers/thoxos