Postern — Audit-Readiness Dossier

Packaging to unblock a contracted external audit — not the audit itself.

`Reference prototype · UNAUDITED · not an audit or assurance`

Postern Labs · 2026-07-12 · for adversarial senior review

Postern — Audit-Readiness Dossier

Purpose: package what a contracted third-party security auditor needs to *scope and begin* an external audit of the Postern products. It is a readiness deliverable, **not** an audit, an assurance, or an attestation.

Prepared by: Fable-5 security engineering (packaging role) **Date:** 2026-07-12 **Primary repo:** `/Users/tiagoacioli/dev/bloch-protocol` (product workspace) **Companion repo (external, by reference):** `bloch-blockchain` / `BlochSISPoW-project` (`gitlab.com/blochsispow-group`) — the ownerless base protocol, consumed here as a pinned git dependency.

---

0. Honest posture (read before anything else — binding rails)

These statements are load-bearing. An auditor should treat any Postern surface that contradicts them as itself a finding.

- **No external audit has been performed on any Postern product or on the base protocol.** An external audit is *contracted but not delivered*. Everything here is "internally reviewed + tests pass," which is explicitly **not** an audit. This document exists to *unblock* that audit, not to substitute for it.
- **Nothing is certified, "secure," "attested," "isolated," "private," or "quantum-safe."** Each such word is earned only by a named artifact + a passed gate, and today none of those gates has cleared. "Mainnet beta" is a designation, not a security claim.
- **The base protocol is trivially forgeable right now.** It runs the relaxed **k=4** Module-SIS PoW regime; the k=8 hardening was activated once at block 213,000 and reverted after a ~4096× difficulty spike stalled the low-hashrate chain. The network is nascent, low-hashrate, and 51%-attackable. No security property attaches to any `k` today.
- **The coin is not a security and not an asset.** No token sale, no listing, no price, no custody. The base protocol is ownerless; nobody (including Postern Labs) controls it. A **17% founder premine** (10-year cliff, 40-year vest, structurally passive) is disclosed in full. No security control anywhere depends on holding or spending the coin. *Do not attach value to the coin.*
- **Key storage is software, not hardware.** Across vault, wallet, keys, and panic lock, keys derive from Argon2id and live in zeroized RAM buffers. StrongBox / Secure Enclave / TEE key-wrapping is planned app-layer work with **no code shipped**.
- **Reproducible-by-design ≠ reproduced. QEMU ≠ hardware. Signed ≠ audited. Tamper-evident ≠ tamper-proof. By-construction isolation ≠ a kernel/hypervisor boundary.** Each distinction is honored in code comments and must be preserved.
- **Naming discipline:** the scheduler daemon is **Yagabona**, never "Baba Yaga" (that word mark is third-party registered). "**Postern OS**" refers *only* to the Nix-built, dm-verity-sealed, attestable images; the postmarketOS overlay and the Android/Waydroid compatibility tiers are **non-attestable community/compat editions** and must never be called "Postern OS."

---

1. Scope & architecture (for the auditor)

1.1 The layered model — ownerless base ⊥ owned products

Postern is deliberately split into two trust layers that meet at a single, pinned cryptographic seam:

The base protocol is a **neutral commons** with no owner, foundation, or official site — just software and a documented RPC/API. The products are **owned by Postern Labs**, rebranded away from "Bloch," and permissively licensed (MIT / Apache-2.0 / BSD; never GPL/AGPL embedded). This is the "BlackBerry model": customers pay for protection, support, and provable integrity — never for a coin.

1.2 The two shared hubs (highest audit leverage)

Nearly everything routes through two shared primitives — audit these first because their blast radius is the whole suite:

1. `bloch-crypto` **hybrid SIGN = ML-DSA-65 ‖ Falcon-1024 (AND-combiner).** The single most-reused primitive: base-chain consensus/RPC/transport identity, Consiglio login verify, Seal attestation verify, and the mobile/desktop `WalletCore`. It lives in the **external** base-protocol repo and is pinned here by git rev `dd8df3526518ba792057708a6aa0efd5a91d8e40`. `mobile/core::WalletCore` is byte-identical to the node by construction, pinned by a golden test. Excellent for auditability (one implementation), dangerous for blast radius (one flaw cascades; there is no crypto-agility/suite-id migration path today).
2. `postern-core` **seal = key-committing XChaCha20-Poly1305 + Argon2id KDF.** The product-side hub; consumed by vault, courier, panic, seal-companion, telephony, and the pairing channel. One canonical wire format and parameter set so no product can drift.

Two secondary shared primitives: the **SHAKE-256 hash-chained audit trail** (`postern-trail` , consumed by yagabona + ezekiel1) and the **fail-closed scoped-expiring-consent kernel** (`postern-policy`).

1.3 Delivery surfaces

- `postern-sdk` — one UniFFI `cdylib` facade, one `setup_scaffolding!`, one Kotlin/Swift binding folding wallet + vault + courier + seal + hygiene + panic + keys for mobile + desktop. Delegates to pure cores; reimplements no crypto.
- `desktop/` (Tauri) and `mobile/` (UniFFI) apps.
- `apps/` — static, CSP-strict, dependency-free single-file HTML consoles per product (consiglio, ezekiel1, izbushka, sotto, vedetta, tolmach, store, login, …).
- `bp-wt-browser` — a hardened **Firefox ESR (Gecko)** distribution (Mullvad Browser / LibreWolf class), *not* a Chromium fork or from-scratch engine.
- `os/` — Nix profiles: `attested.nix`, `seal-gate.nix`, `cloud.nix`, `bloch-node.nix`, `browser.nix`, `desktop.nix`, `mobile.nix`.
- `enterprise/` — cloud control-plane + hosted services, comms app, shield (sealed container / DLP), and the Ed25519 licensing chain.

1.4 Trust boundaries

Boundary	What crosses it	Trust assumption
Base ⊥ products	Signed txs, block state, `bloch-crypto` code (pinned rev)	Base is forgeable today; products inherit a **zero-security floor** until k=8 + audit + hashrate
Device ⊥ device (pairing/signer)	Hybrid-PQ session frames, QR/SAS confirmation	Human out-of-band SAS anchors against MITM; ml-kem leg unaudited, X25519 is the safety leg
Client ⊥ hosted cloud (enterprise)	Opaque ciphertext, license tokens, node descriptors	**Hosted = less private than self-hosting** (stated first, in code); relay never holds plaintext/keys; a compromised control plane defeats by-construction isolation
Userland process ⊥ everything else	Governance decisions, sealed data, audit trail	Integrity **by construction** in a trusted process — a real correctness property, **NOT** a kernel/hypervisor boundary
Software keystore ⊥ host OS	Derived keys, wallet seed	A host/OS compromise reads the derived key or seed — no hardware isolation today
Guest VM ⊥ host (OS attestation)	dm-verity roothash, SEV-SNP/TDX quote	Aspirational: profiles unbuilt on real hardware; QEMU ≠ hardware

---

2. In-scope vs out-of-scope for *this* audit

2.1 In scope (the owned products, this repo)

- `crates/postern-core` — the seal (key-committing XChaCha20-Poly1305), Argon2id KDF, attestation transcript.
- `crates/postern-{vault,courier,messenger,keys,panic,seal-companion,hygiene}` — product cores.
- `crates/postern-{consiglio-auth,trail,policy}` — auth challenge/response, the SHAKE-256 audit trail, the fail-closed consent kernel.
- `crates/{ezekiel1,yagabona}` — access-governance engine + scheduler daemon.
- `integration/{pairing,pairing-transport-lan,signer-protocol}` — device pairing and the phone-as-air-gapped-signer protocol.
- `enterprise/{cloud,comms,shield,licensing,integration}` — hosted control-plane + services, the enterprise comms app, the sealed shield container, the Ed25519 license chain.
- `os/*.nix` node-side attestation code (roothash read/report/verify) + the seal companion verifier.
- `bp-wt-browser` — the hardened Firefox distribution (policies.json, user.js, nix derivation).
- `desktop/` (Tauri) integrity pin + CSP.

2.2 Out of scope for this engagement

- **The ownerless base protocol (Bloch-SIS-PoW).** Consensus (GhostDAG-Q, reorg), the Module-SIS PoW gate and its hardness, hybrid SIGN internals, the Coherence shielded pool, FFG/DKG, and Stratum are a **separate audit** against the external repo. They are summarized here (§4.6) only as the trust floor under the products. The base repo has its own internal audit history (see §4.6) and its own external-audit plan (candidate firms: Kudelski, NCC Group, Trail of Bits, Certik, Halborn).
- `ffg/` **and the RWA/FFG work** — these are **designs**, not built code (see the FFG-RWA design + independent safety review). No implementation to audit yet.
- **The** `bloch-crypto` **git-dep internals** (Falcon/ML-DSA composition, address derivation) — external code, pinned by rev; belongs to the base-protocol audit, but the auditor must be told the products' PQ signature claims *rest on it*.
- **Third-party upstreams** consumed as crates (see §3 "audited-upstream" column) — trusted per their own posture; listed, pinned, and license-gated, not re-audited.
- **The known relaxed-k PoW forgeability** — a documented, intentional gate, not a bug. An exploit that persists *after* k=8 + difficulty fix would be in scope for the base audit.

---

3. Cryptographic inventory

Every primitive, where it is used, its upstream provider + pinned version, and its **assurance status**: *audited-upstream* (mainstream crate with external review history — Postern's *use* still needs review) vs *unaudited-in-Postern* (Postern-written or pre-1.0/unaudited upstream).

3.1 Symmetric / AEAD / KDF / hashing

Primitive	Exact construction & params	Where used	Upstream + version	Assurance
XChaCha20-Poly1305 key-committing seal (v2)	Wire `0x02 ‖ nonce(24) ‖ commit(32) ‖ ct`. Per-seal subkey via **HKDF-SHA-256** salted by nonce; `commit` is an HKDF output stored & **constant-time checked before decrypt** (closes invisible-salamander / partitioning). Context string binds into HKDF `info` **and** AEAD AAD (cross-product substitution refused). Legacy `nonce‖ct` read-only for back-compat.	`crates/postern-core/src/lib.rs`; every seal consumer (vault, courier, panic, shield, pairing channel, comms sealed-store)	`chacha20poly1305` 0.10.1, `hkdf` 0.12.4, `sha2`, `subtle` 2.6.1 (CT compare), `zeroize` 1.9.0	Construction is AWS-Encryption-SDK-style, correct; **use is unaudited-in-Postern**. Tests `commit_tamper_is_detected`, `context_binding_is_enforced`, `wrong_key_fails_to_open` (11/11).
Argon2id KDF	Two pinned tiers: **at-rest** m=64 MiB, t=3, p=1, v0x13 (RFC-9106 profile); **floor** m=19 MiB, t=2, p=1 (library default, pinned so a bump can't silently re-derive).	`crates/postern-core` `kdf::derive_key`; vault passphrase, shield seal, pairing persist	`argon2` 0.5.3 (pure-Rust)	Audited-upstream primitive; **params & `p=1` serial-lane choice need an external eye** (justified in-comment). Low 19 MiB floor on constrained targets is offline-attackable.
HKDF-SHA-256	Domain-separated, distinctly-labeled expand for: per-seal subkeys, courier X-Wing combiner, pairing initial/SAS/resume keys, mutual-confirmation tags.	postern-core, courier, pairing (`resume.rs`, `hybrid.rs`)	`hkdf` 0.12.4	Audited-upstream; use unaudited-in-Postern.
SHA3-256	Domain-separated challenge/login transcript.	`crates/postern-consiglio-auth` (`challenge.rs`, `LOGIN_DOMAIN`)	`sha3` 0.10.9	Audited-upstream; use unaudited.
SHAKE-256	Hash-chained tamper-evident audit trail: `chain_hash = SHAKE256(prev_hash ‖ canonical_json, 32)`, JSONL, per-append fsync, genesis `"genesis"`. Also hygiene `prove` / `seal`.	`crates/postern-trail`; `yagabona` + `ezekiel1` trails; `postern-hygiene`	`sha3` 0.10.9	Tamper-**evident** (not proof): a root actor can rewrite the file and recompute the chain. Note: implemented `chain_hash` currently omits the spec's version/domain tag — flag for the auditor.
AES-256-GCM	(a) **Reference-only** default sealer in comms sealed-store (WebCrypto, **non-committing, no Argon2id**) — explicitly "do not ship as the seal." (b) Wallet-at-rest in the base wallet encryption path.	`enterprise/comms/app/persistence/sealed-store.mjs` (finding COMMS-1); base wallet	WebCrypto / RustCrypto	Reference sealer is a **known prototype seam** (COMMS-1) — production must inject the postern-core seal.

3.2 Post-quantum & hybrid

Primitive	Exact construction	Where used	Upstream + version	Assurance
Hybrid SIGN: ML-DSA-65 ‖ Falcon-1024 (AND-combiner)	Both signatures required; offsets 1952/4032/3309. The one trust-root signature.	`bloch-crypto` (external, pinned rev); consumed by consiglio-auth login, seal-companion attestation verify, wallet signing, hygiene optional seal	`pqcrypto-mldsa` 0.1.2, `pqcrypto-falcon` 0.4.1 (via `pqcrypto-internals`, base-repo git dep)	**Unaudited-in-Postern and external.** Widest blast radius; classic hybrid-composition failure mode (weak binding / cross-scheme confusion) is exactly what the base audit must check. No suite-id / migration path today.
Hybrid KEM: ML-KEM-1024 ‖ X25519 (X-Wing-style)	HKDF-SHA-256 combiner binds full transcript (wire version, KEM ct, recipient key). Adds FIPS-203 §7.2 encap-key canonicality check upstream skips; rejects low-order X25519 points.	`crates/postern-courier` (file/message drop); `integration/pairing` channel key; `integration/signer-protocol` session	`ml-kem` 0.2.3 (**pre-1.0, UNAUDITED**), `x25519-dalek` 2.0.1, `curve25519-dalek` 4.1.3	Genuine hybrid — a break of either leg alone leaves payload protected. **`ml-kem` is pre-1.0 and unaudited; X25519 is the deliberate safety leg.** No sender authentication (courier); size-revealing AEAD.
Hybrid KEM (transport, base): ML-KEM-768 / Kyber768	Node P2P handshake KEM. Self-labeled "properties NOT claimed," and per the base audit **not yet fully wired into libp2p**; libp2p identity remains **Ed25519 (classical)**.	base protocol transport (out of scope here)	`pqcrypto-kyber` 0.8.1	Base-protocol scope. Note the KEM≠signature gap: transport confidentiality is PQ, peer *authentication* is classical Ed25519.
Megolm / Olm (vodozemac)	Matrix group/1:1 ratchet, **classical Curve25519** — no PQ on transport. Behind an **off-by-default** `matrix` feature. A hand-rolled `megolm_reference` is legacy/reference only, not wired to transport.	`crates/postern-messenger`	`vodozemac` 0.10.0 (Apache-2.0, audited by Least Authority 2022), `matrix-sdk` 0.18.0	Audited-upstream primitive; **Postern integration unaudited**. Harvest-now-decrypt-later applies to intercepted transport; in-memory store only (no persistence yet).
WebAuthn ES256 / P-256 (passkey core)	Per-relying-party `SigningKey` (`ZeroizeOnDrop`, compile-asserted); private key exported only to be sealed in the vault; `to_vault_record` binds key ↔ rp_id ↔ credential_id atomically.	`crates/postern-keys` (FIDO2 authenticator core)	`p256` 0.13.2, `ecdsa`	Classical (no PQ passkeys until relying parties accept them). **Software authenticator** — weaker than a hardware key; it is a *core*, not a finished CTAP2/WebAuthn product.

3.3 Signatures & attestation (product side)

Primitive	Construction	Where used	Upstream + version	Assuranc
Ed25519 license chain	Detached Ed25519 (Node `node:crypto` built-in, raw 32-byte keys, versioned token `v:1`) over **canonical-JSON (sorted keys)**. Chain: VENDOR key → ORG **master** token (DUNS-scoped or freemium) → **sub** tokens signed by org master. `verifyChain` re-verifies each link offline against the pinned vendor pubkey + clock; authenticity and expiry reported separately; fail-closed on parse/sig/DUNS/expiry.	`enterprise/licensing/src/*.ts`; `enterprise/integration/license-adapter.mjs` `verifyChain` — the single authority seam for console, comms sub-access, shield sub-token	`node:crypto` (zero third-party crypto dep)	**Unaudit scaffold.** a **DEV S** `vendor.` productio **revocati anywhe** expiry-on **only (ho disclose** Dun & Br `VENDOR_` currently in the hos SVC-1).
SEV-SNP attestation verifier	Pure-Rust (`sev` 8.0.0, features `snp` + `crypto_nossl`; p384 ECDSA + RSA-PSS) chains VCEK → ASK → ARK to AMD's **pinned** roots (Milan/Genoa/Turin, never taken from the wire); checks measurement, HOSTDATA policy hash, `report_data = SHA-256(nonce ‖ guest_pubkey)`, min-TCB anti-rollback floor, VMPL, refuses debug guests. Default (non-SNP) build fails closed to `Unverified` without an out-of-band anchor. TDX is a documented fail-closed **stub**.	`crates/postern-seal-companion/src/{snp,binding,tdx,gate}.rs` (feature `sev-snp`, off by default); `os/seal-gate.nix`, `os/cloud.nix`, `os/attested.nix`; `os/ATTESTATION.md`	`sev` 8.0.0, `p384` 0.13.1, `rsa` 0.9.10; `bloch-crypto` for the PQ sig leg	**Unaudit** (a verifier forgeries) **revocati** 2020-era VCEK wo SNP off b TDX/And **exercise from ha launche** "unpinne any self-s review fin **in code** (`Unverif`
Device pairing / air-gapped signer transcript	QR/SAS-anchored ML-KEM-1024|X25519 session (6-digit ~20-bit SAS on a **separate HKDF branch**; explicitly **not a PAKE**); XChaCha20-Poly1305 frames with **monotonic per-direction counter anti-replay** + session-id in AAD + direction labels (anti-reflection); forward-secret re-handshake on resume; WYSIWYS domain-separated transcript `postern:signer-protocol:v1` signed over a SHA3-256 digest; sign-after-commit ordering; secret keys structurally non-serializable (compile-time denylist test).	`integration/pairing`, `integration/pairing-transport-lan`, `integration/signer-protocol`	ml-kem 0.2, x25519-dalek 2, chacha20poly1305 0.10, hkdf 0.12, sha3 0.10	Self-label **unaudit** only if the performe to a full re `sessio` **`signer`: is a dete stand-in cryptogr** is `mobile/` (hybrid F LAN/mD unauthen
Desktop node integrity pin	Plain software **SHA-256** over the binary — stated verbatim as "not a signature, not hardware attestation, not a TEE check." Fail-closed on mismatch.	`desktop/` (Tauri)	RustCrypto sha2	Correctly depth, no
Release signing	`minisign`-signed `SHA256SUMS`; public key `RWTSe98wfKKHuVRDnywuQmH/cMyP/HNuD/hFFLQeKdQeAHENliqJpmKs`, published in ≥3 independent places.	Release artifacts / distribution	minisign	Signed ≠ image att security.

3.4 Base-protocol primitives (out of scope here, listed for completeness)

- **Module-SIS PoW gate** — SHAKE-256 hashcash + Module-SIS structural gate; `k=4` relaxed regime today (forgeable), `k=8` canonical (reverted). No external reference to differ against — self-referential KATs only.
- **Coherence privacy layer** — shielded pool + spend proofs; the **SP1 / FRI** zero-knowledge shielded-tx work is a **scaffold** in the base repo (`docs/specs/COHERENCE-v0.2.md`), not a product core in this tree.
- **GhostDAG-Q consensus, Dandelion++ metadata privacy, FFG/DKG** — base repo.

---

4. Known threat model & self-identified weaknesses

Postern's honesty discipline is unusually strong: nearly every weakness below is **already stated in the code or a review doc**. The audit's job is to *test* these, find the *undisclosed* ones, and convert self-review into third-party assurance.

4.1 Enterprise / cloud / shield (from the Cloud+Enterprise Security Review)

ID	Sev	Where	Weakness	Required before production
SEAL-1	HIGH (design contract)	`enterprise/shield/container/src/grant-binding.mjs:104-107,155-162`	"revoke ⇒ inert" rests on `grantSecret` being ephemeral/broker-released per session and **never persisted**; nothing enforces or documents that. A deployment caching it in local storage breaks revoke for even a non-rooted ex-employee.	Mandate broker-per-session fetch, never on disk; make it a first-class deployment invariant.
SEAL-2	MED	`grant-binding.mjs:140-144`	`revoke()` flips the flag but does **not** zeroize `grantSecret`; bytes linger until GC (RAM inspection just after revoke still finds it).	Auto-zeroize on revoke/expiry (free fix).
SEAL-3	HIGH (prod)	`real-seal-engine.mjs:96-115` + `seal/src/main.rs`	The 32-byte key crosses a **process boundary as hex** (stdin for seal/open; **stdout** for `derive-key`). A supervisor/log pipeline capturing stdout leaks the key. Labeled honestly as prototype wiring.	Production must use in-process **FFI/WASM** so the key never crosses a boundary.
COMMS-1	MED	`enterprise/comms/app/persistence/sealed-store.mjs:77-112`	Default `referenceSealer` is WebCrypto **AES-256-GCM** — non-committing, no Argon2id — explicitly "do not ship as the seal."	Inject the real postern-core seal via `opts.sealer`.
SVC-1	MED	`enterprise/cloud/services/src/console-service.mjs`	`VENDOR_PUBLIC_KEY` is **optional/fail-open**; unset ⇒ uploaded master tokens accepted **unverified** ⇒ a forged workspace registers.	Make mandatory/fail-closed in production.
SVC-2	MED	relay + console services	`ALLOWED_ORIGIN` defaults to `*`.	Require explicit origin in production.
CP-1	LOW	`enterprise/cloud/control-plane/src/tenant.mjs:115-141`	`tenantId` is 64-bit **FNV-1a** (non-crypto) — an isolation label only, but has a birthday bound.	Optional: truncated SHA-256.

Positives verified: cross-tenant leakage refused *by construction* (per-tenant scoped `Map`, namespaced, no cross-tenant query API); revoke ⇒ deprovision wipes the whole namespace; the relay is transport-only and never holds plaintext or keys; the honest "hosted = less private" / "by-construction ≠ kernel" / "unaudited" disclosures are in the code.

4.2 Product cores (from the Products Security/PQ audit)

- **Software keystore everywhere.** Vault/wallet/keys/panic derive keys from Argon2id into zeroized RAM; **no** TEE/StrongBox/Secure-Enclave wrapping ships. A stolen sealed vault + captured passphrase = game over (offline resistance is Argon2id only). This is the single largest substantive ceiling — disclosed, unbuilt.
- **Courier has no transport and no sender authentication.** The crypto (hybrid PQ seal) is sound; the planned ephemeral Tor onion *service* (arti) is NOT implemented — today it seals with nothing to transmit over. A valid open proves nothing about who sealed. Key authenticity is out-of-band (fingerprint compare, or an active MITM reads everything). No padding ⇒ size/timing/endpoint metadata leaks.
- **Messenger transport is classical + non-persistent + non-Tor.** PQ only at rest (via the vault); harvest-now-decrypt-later applies to transport; in-memory store loses session/replay state on restart; social-graph metadata visible to the homeserver.
- **Keys is a core, not a product.** Full CTAP2/WebAuthn + browser/Android integration is unbuilt; no hardware-backed storage.
- **Panic lock is *not* a duress/deniable tool for the primary lock** (it forbids those words for itself — the locked state is observable, no decoy, a coercer can compel the one real key). The separate deniable container is single-snapshot only — **cloud/ multi-snapshot backups break it** (the sharp edge).
- **Seal-companion trust roots are mostly out-of-band pinned keys.** Only SEV-SNP chains to a hardware vendor root; TDX/Android are stubs. Do not describe any device as "attested" by default. Attestation proves the *build*, not the absence of a kernel 0-day or a coerced user.
- **PQ signature stack is an external, pinned, unaudited git-dep** (`bloch-crypto`) — the wallet/attestation PQ claims rest on code outside this tree.
- **Argon2 `p=1` + 19 MiB floor tier** on constrained targets — confirm the memory-hardness envelope for offline-attackable at-rest secrets.

4.3 Browser (`bp-wt-browser`) — self-identified inconsistencies

Hardened Firefox ESR; honesty labelling is excellent ("HARDENED, NOT ANONYMOUS," "NOT Tor," "UNAUDITED," `build-manifest.json` carries `reproducible=false attestable=false audited=false status=BUILT`). Real ship-blockers to reconcile:

1. **DNS posture self-contradiction.** `user.js` hard-sets `network.trr.mode=5` (DoH OFF) and claims the resolver is an unmade founder decision, but `policies.json` + `postern-browser.nix` ship **Mullvad DoH ON, Locked** — a default build fails its own `VERIFY.md` checklist.
2. **`Fallback:true` on the DoH policy** can leak to plaintext system DNS — the exact failure `user.js` warns about. Remove (hard-fail) or explicitly disclose.
3. **uBlock install-mode label mismatch** (`force_installed` vs `VERIFY.md`'s `normal_installed`).
4. **nixpkgs pin is a `TODO(pin)`** in `browser/postern-browser.nix` — the build falls back to ambient `<nixpkgs>` with a loud trace ("NOT rebuild-verifiable until the pin is filled"). Not reproducible until pinned.
5. **Branding ship-blocker:** the de-brand is external only and does **not** rewrite internal "Firefox"/"Mozilla" strings — builds from this file are flagged **DEV-ONLY, MUST NOT be published** until it lands.

Note: the browser repo also carries formal per-review self-audits in `bp-wt-browser/docs/reviews/` (finding series **F1–Fn**, e.g. crypto-core F1–F12, messaging-tor F1–F13, an SDK KDF-parameter-downgrade where `kdf.iter` is rewritable to `1`, the Consiglio governance-key sealing review). These are the code-side self-identified findings and are useful auditor input alongside the counselor review docs.

4.4 OS attestation & reproducibility

- **Attestation is aspirational.** `attested.nix` / `cloud.nix` / `seal-gate.nix` are **unbuilt profiles** "written on a non-Nix host — NOT built here." Nothing is "attested" until it boots on real SEV-SNP/TDX hardware against `/dev/sev-guest`. SEV-SNP off by default, TDX a fail-closed stub, no VCEK revocation. **QEMU ≠ hardware.** The **seal-gate** boot design (codename Chiostro, `os/cloud.nix`, port 16510) is *attestation-before-connect*: it is the only pre-attestation listener; per connection it binds a fresh nonce into `report_data = SHA-256(nonce ‖ WG_guest_pub)`, and the WireGuard tunnel opens only after `Verdict::Trusted`. The verifier crypto + wire format and the node-side roothash read/report/verify are implemented and vector-tested; the `/dev/sev-guest` path is not exercised on real hardware. DLP audit trail is SHA-256 hash-chained (tamper-evident, not tamper-proof).
- **Reproducibility is FALSE until measured.** No `flake.lock` committed and `mobile-nixos` is a floating branch ⇒ "same input → same image" is not demonstrated. Committing the lock buys *"reproducible-by-design,"* not *"reproducible"* — the word is earned only when two independent builders emit a **bit-for-bit identical, diffoscope-clean** image. **Reproducible-by-design ≠ reproduced.** `os_roothash` is a design reference, not yet a verified value.

4.5 Naming / tier-bleed (the named #1 systemic risk)

"Trust-label bleed" — an `attestable` / `verified` / `secure` / `private` / `isolated` badge rendering on a surface that hasn't earned it — is the single failure that poisons the whole promise. Mitigation is a shared `EnforcementMode` /tier enum + one badge component + a CI assertion. Auditor should check that no console rounds up. The postmarketOS and Android/Waydroid tiers are **non-attestable** and must never read "Postern OS."

4.6 Base-protocol trust floor (separate audit — context only)

The products sit on a base that is forgeable today. The base repo's own review history:

- **Launch-day audit (2026-04-20):** 3 Critical / 6 High / 11 Medium / 5 Low. All 3 Criticals since **resolved** (UTXO reorg handling C-1, deterministic seed keygen C-2, duplicate-tx merkle CVE-2012-2459 C-3); ~40% of the total closed as of 2026-04-21. Remaining High/Medium (e.g. libp2p identity still Ed25519, `past_blue_set` bound, treasury single-sig) pending.
- **Progressive k-ramp consensus review (2026-07-12): NO-GO** as specified — the difficulty engine saturates so the ramp is structurally blocked, and each step is a bilateral hard fork (not a soft fork). No security property attaches to any `k`.
- **FFG/RWA independent safety review:** the FFG finality gadget is **design only** and, as scoped, is a **Postern-operated permissioned notary checkpoint**, *not* Byzantine finality ("BFT over one org's nodes is not BFT"). Ceiling = `min(base security, Postern integrity)` = zero today. Do not label base-forgeable checkpoints "HardFinalized" to integrators.
- An **internal pre-mainnet audit plan** exists (5 passes: boundaries, state machines, crypto correctness, consensus invariants, adversarial) and names candidate external firms — that is the base-protocol external engagement, distinct from this products audit.

---

5. Build / reproducibility story (what an auditor needs to reproduce the bytes)

5.1 Supply chain (already in place)

- **Rust:** `Cargo.lock` committed; the PQ-crypto fork is vendored/pinned (`pqcrypto-internals` + `bloch-crypto` pinned by git rev `dd8df3526518ba792057708a6aa0efd5a91d8e40`) so the workspace is self-contained.
- **`cargo deny check`** (`deny.toml`) enforces permissive licenses (no copyleft), no external git deps beyond the pinned base, and RUSTSEC advisories — run in CI.
- **`gitleaks`** secret scanning in CI; **fuzz** targets for untrusted-input parsers.
- **Websites** load zero third-party scripts/styles/fonts/trackers; strict CSP.

5.2 What must be frozen BEFORE the engagement

1. **Commit `flake.lock` + pin `mobile-nixos` rev.** Non-negotiable: without it "the audited artifact" is undefined; the auditor must `nix build` the *exact bytes*.
2. **Freeze the crypto spec + parameters** (base repo: `params.rs`, the KAT regression vectors, `docs/specs/POW-HARDNESS.md`, `BLOCH-SIS-ATTESTATION.md`, `COHERENCE-v0.2.md`). Note the honest gap: KATs are **regression** vectors ("the crate IS the reference"), **not** standards KATs — precisely what an external cryptographer fills.
3. **Tag a frozen "audit bundle":** one immutable git tag containing frozen spec docs, KAT vectors, `Cargo.lock`, `flake.lock`, and exact crate revs.
4. **Ship this honest scope statement** (§0 + §2).

5.3 Determinism: claimed vs proven

Claim	Status	How it's earned
"Deterministic-by-design / inputs pinned"	Achievable now (commit `flake.lock`)	`nix flake lock`; record pinned nixpkgs + mobile-nixos revs
"Reproducible" (bit-for-bit)	**NOT yet proven**	Single-host `nix build … --rebuild --check` (start native aarch64 `.#packages.aarch64-linux.bloch` + `.#mobile-image` — RocksDB 8.10 C++ is the #1 nondeterminism risk), then **two independent builders** compare `narHash` (+ image sha256 + dm-verity roothash) via `repro-manifest.sh` / `repro-compare.sh`; must be diffoscope-clean
"Attested"	**NOT yet**	Boot `.#attested-image` on real SEV-SNP/TDX (`/dev/sev-guest`) and validate the chain

Do **not** publish `os_roothash` or say "reproducible" until two independent builders match. Until then the honest phrase is **"deterministic-by-design, reproducibility not yet independently verified."**

5.4 Recommended audit sequencing (highest trust-per-dollar first)

1. **Base:** hybrid PQ signatures + address derivation composition, then Module-SIS PoW soundness/hardness (no external reference exists — this is where independent review is irreplaceable).
2. **Products/attestation:** the seal-companion SEV-SNP cert-chain verifier + report binding (a verifier bug silently accepts forgeries).
3. **Products:** the postern-core seal, courier hybrid KEM (unaudited `ml-kem`), pairing/ signer protocol, enterprise gates (SEAL-1/3, SVC-1).
4. **Hardware validation** (not an audit): boot the attested image on real CVM hardware.

---

6. Auditor checklist

Onboarding / scoping - [ ] Confirm the frozen audit-bundle git tag + committed `flake.lock`; `nix build` the exact reviewed bytes. - [ ] Confirm the base-protocol audit is a **separate** engagement; obtain the pinned `bloch-crypto` rev and its spec/KAT bundle. Resolve the cosmetic inconsistency where several manifests carry a `<A_MERGED_SHA>` placeholder narrative in comments while the actual `rev =` is the real SHA `dd8df35…` — confirm the pin is the intended one. - [ ] Read §0 rails and flag any product surface that contradicts them (that is itself a finding).

Crypto correctness (product side) - [ ] postern-core seal: key-commitment CT-check-before-decrypt, context binding in both HKDF `info` and AEAD AAD, nonce handling, ZeroizeOnDrop assertions. - [ ] Argon2id params (64 MiB/t3/p1) and the 19 MiB floor tier — memory-hardness envelope for offline-attackable secrets; the `p=1` serial-lane choice. - [ ] Courier hybrid KEM: X-Wing transcript binding, FIPS-203 §7.2 canonicality check, low-order X25519 rejection, zeroization with no early-return gaps; the pre-1.0 `ml-kem` leg. - [ ] Pairing/signer: SAS anchoring vs active MITM, monotonic-counter anti-replay, session-id-in-AAD cross-session replay barrier, forward-secret resume fail-closed. - [ ] consiglio-auth: SHA3-256 transcript domain separation, challenge nonce/expiry, replay, delegation to `bloch_crypto::verify`. - [ ] Ed25519 license chain: canonical-JSON determinism, offline chain verification, clock handling, DUNS binding, revocation model, sub-subject match; **fail-open `VENDOR_PUBLIC_KEY` (SVC-1)** and the dev sample key. - [ ] Seal-companion SEV-SNP: VCEK→ASK→ARK to pinned AMD root, fresh-nonce binding, fail-closed without an out-of-band anchor, TDX/Android stub behavior. - [ ] SHAKE-256 trail: canonical-JSON, missing version/domain tag, tamper-evident vs tamper-proof boundary, standalone verifier.

Enterprise gates (production-acceptance deltas) - [ ] SEAL-1: `grantSecret` broker-per-session, never persisted (enforced/documented). - [ ] SEAL-3: replace CLI process-boundary key handling with in-process FFI/WASM. - [ ] COMMS-1: postern-core seal injected; AES-GCM reference sealer never shipped. - [ ] SVC-1/SVC-2: mandatory `VENDOR_PUBLIC_KEY` + explicit `ALLOWED_ORIGIN`. - [ ] Cross-tenant isolation is by-construction (no cross-tenant query API); revoke ⇒ deprovision wipes the namespace; relay holds no plaintext/keys.

Keystore / hardware - [ ] Confirm no shipped path claims hardware-backed keys; software keystore threat model (host/OS compromise reads seed/derived key).

Browser - [ ] Reconcile the DNS contradiction (user.js vs policies.json vs nix); DoH `Fallback` leak; uBlock install-mode label; confirm no anonymity/attestation overclaim.

Build / repro / attestation - [ ] `flake.lock` committed + drift gate in CI; run `--rebuild --check`; run the two-builder `narHash` comparison; confirm nothing is called "reproducible" or "attested" prematurely. - [ ] No AGPL/GPL in the image closure (dev-only GPL tooling like diffoscope excluded).

Naming / claims - [ ] No tier-bleed: `attestable/verified/secure/private/isolated` render only on earned surfaces; community/pmOS/Android tiers never called "Postern OS"; scheduler is "Yagabona" never "Baba Yaga"; coin never framed as a security.

---

7. Open questions to resolve before / during the audit

1. **`bloch-crypto` hybrid-SIGN composition** — is the ML-DSA-65‖Falcon-1024 AND-combiner binding sound against cross-scheme confusion, and is there *any* crypto-agility/ suite-id migration path (today algorithm identity is implicit in byte offsets)?
2. **`ml-kem` pre-1.0 risk** — acceptable while X25519 is the safety leg, but what is the upgrade plan when `ml-kem` reaches a stable audited release?
3. **Argon2 envelope** — is `p=1` + a 19 MiB floor defensible for offline-attackable at-rest secrets, given no hardware key binding?
4. **Software-keystore ceiling** — when (if) does TEE/StrongBox/Secure-Enclave wrapping land, and how is the interim host-compromise risk communicated to users?
5. **SEAL-1 deployment invariant** — will `grantSecret`-never-persisted be enforced in code or only documented?
6. **License revocation** — beyond token expiry + tenant deprovision, is there an active revocation/CRL mechanism? What replaces the dev sample vendor key, and how is the production vendor key custodied?
7. **Attestation trust-root distribution** — how are out-of-band anchors distributed for pinned-device/TDX/Android paths (the operational risk that gates the word "attested")?
8. **Reproducibility ownership** — who runs the two-builder check on which hardware, and is the aarch64 `.#attested-image` variant added so the sealed image repro-checks natively?
9. **Trail version tag** — will the SHAKE-256 `chain_hash` adopt the spec's domain/version tag before format bumps make it ambiguous?
10. **Base ⊥ products gate** — the products' security floor is zero until k=8 is live + stable + hashrate-sufficient + the base audit is *performed*. What is the explicit ordering, and does any product surface currently imply otherwise?

This dossier is an audit-readiness package only. It is not an audit, a security guarantee, a certification, or an attestation of any Postern product. No external audit has been performed. The base protocol is forgeable today; the coin is not a security and has no value; key storage is software, not hardware; nothing is "reproducible," "attested," "isolated," "private," or "secure" until its specific gate clears.