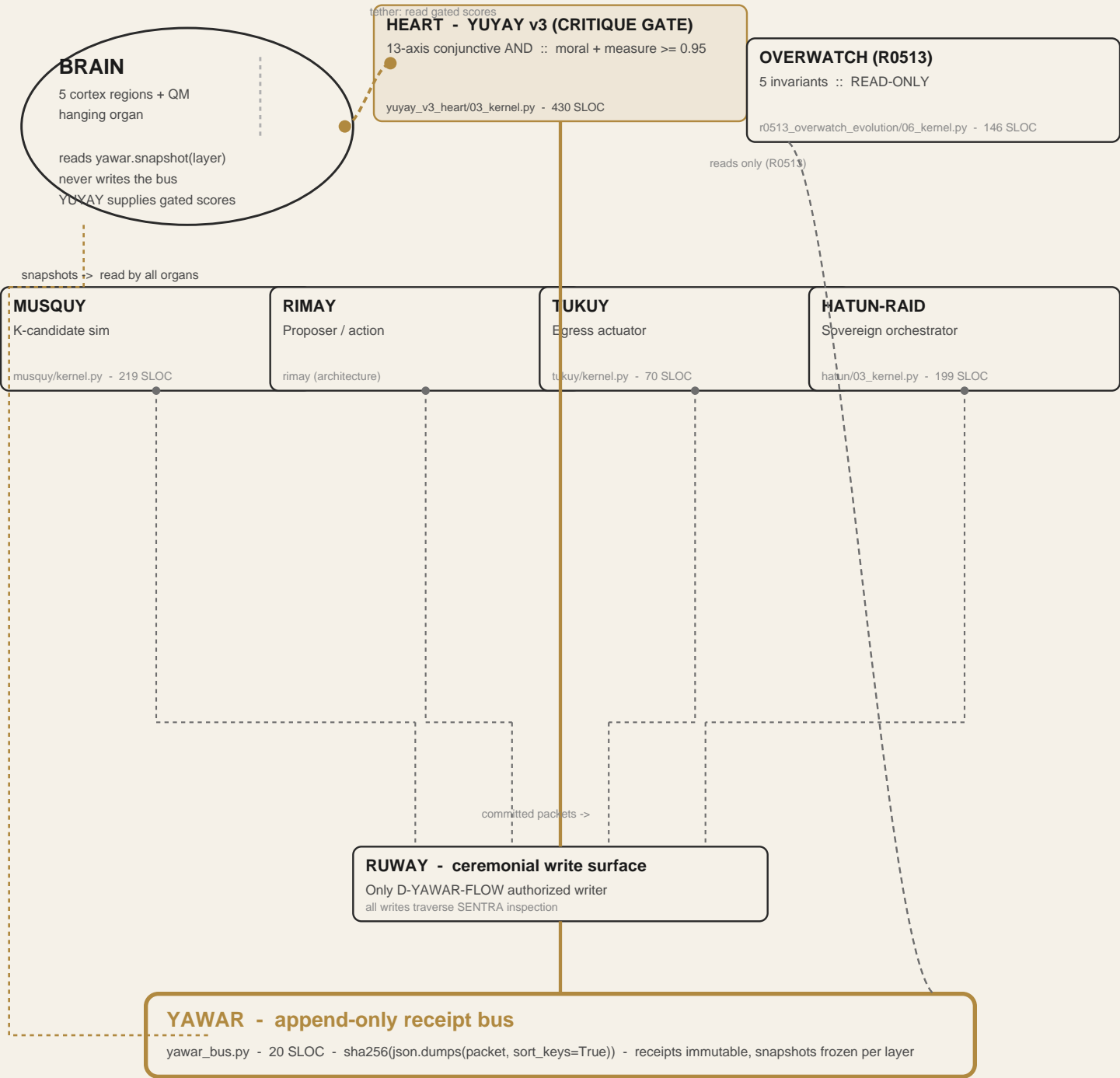


BLOOD — CIRCULATORY WIRING (rundown)

How YAWAR connects every organ. Append-only SHA-256 receipts. Brain hangs off the bus by a single tether: it READS snapshots, never WRITES.



WIRE LEGEND

- gold solid = write path (heart -> RUWAY -> YAWAR)
- gold dashed = read tether (brain + every organ snapshot)
- slate dashed = candidate flow into RUWAY / R0513 read-only audit

BLOOD — DEPTH (receipt anatomy, append + snapshot internals)

What one receipt actually is, on disk. The full source of yawar_bus.py is 20 SLOC — every guarantee below is read straight from it.

yawar_bus.py - 20 SLOC (verbatim, on disk)

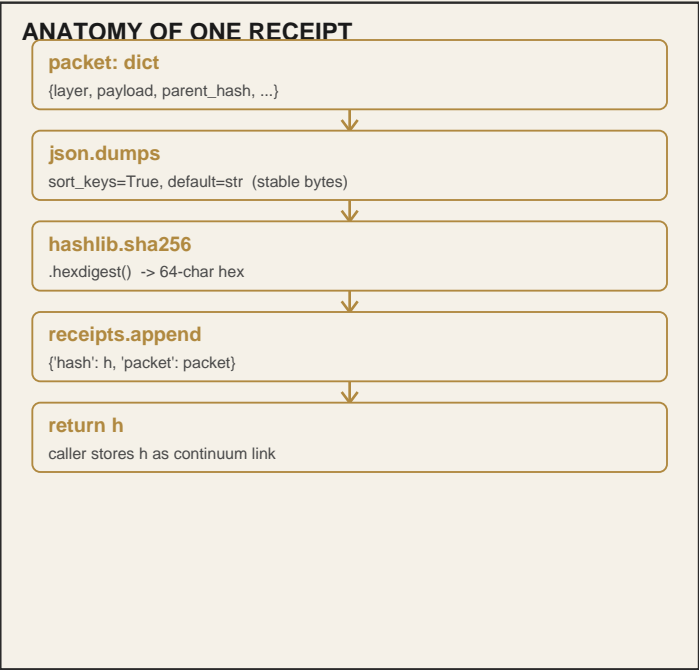
```
"""YAWAR - blood / receipt bus. Append-only, immutable."""
import hashlib, json
from typing import Any

class Yawar:
    def __init__(self):
        self.receipts = []
        self.snapshots = {}

    def append(self, packet, sentra_inspect=None) -> str:
        if sentra_inspect and not sentra_inspect(packet):
            raise PermissionError("SENTRA rejected packet")
        h = hashlib.sha256(
            json.dumps(packet, sort_keys=True,
                        default=str).encode()).hexdigest()
        self.receipts.append({'hash': h, 'packet': packet})
        return h

    def snapshot(self, layer, data):
        self.snapshots[layer] = json.loads(
            json.dumps(data, default=str)) # frozen copy

    def read(self, layer):
        return self.snapshots.get(layer, {})
```



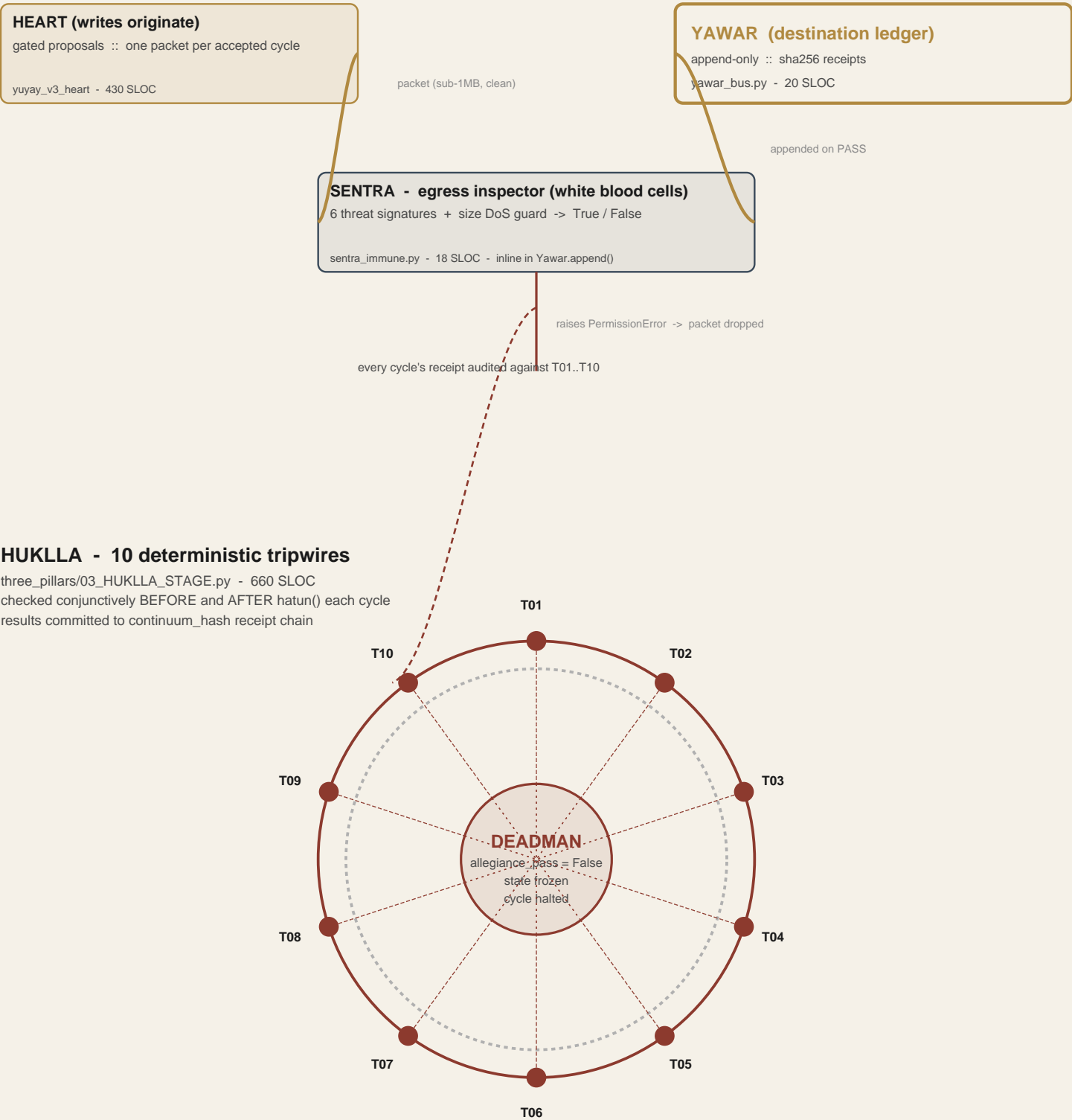
GUARANTEES (read directly from yawar_bus.py source)

- Append-only: self.receipts.append(...) — no method exists to delete or mutate prior receipts.
- Stable serialization: json.dumps(..., sort_keys=True, default=str) -> deterministic bytes for any packet whose values are timestamp-free.
- Cryptographic link: hashlib.sha256(..).hexdigest() -> 256-bit, collision-resistant under standard assumptions.
- Inline immune check: Yawar.append calls sentra_inspect(packet) BEFORE compute — refused packets raise PermissionError; never reach the ledger.
- Frozen snapshots: snapshot(layer, data) deep-copies via json round-trip — readers cannot mutate the live source object.
- Read tether: read(layer) returns the dict or empty — never raises, never blocks (D-YAWAR-FLOW read side).

Honest scope: the receipt is byte-identical across re-runs ONLY when packet values are timestamp-free.
Receipts that include _iso_now() (e.g. musquy/kernel.py) are intra-run deterministic only. Documented in musquy/00_RESULT.md after audit.

IMMUNE — WIRING (rundown)

Two layers in series. SENTRA inspects every outbound packet inline. HUKLLA fires the deadman if any of 10 invariants trip. No partial halt. No override.



IMMUNE — DEPTH (tripwire ledger + SENTRA source)

Every row below is a fixture read directly from on-disk source. No paraphrase. No invention.

HUKLLA — canonical T01..T10 (D45, source: three_pillars/03_HUKLLA_STAGE.py)

Source: TRIPWIRE_DESCRIPTIONS dict, lines ~85-95. Exact strings retained.

#	ID	CONDITION
T01	MORAL_GROUNDING	moralGrounding axis < 0.95 — hard fail, no override
T02	MEASURABILITY_HONESTY	measurabilityHonesty < 0.95 — hard fail, no override
T03	EVAL_AWARENESS	introspection contains eval-awareness flag (NLA/Apollo)
T04	DECEPTION_KEYWORDS	introspection contains 'manipulate' / 'avoid detection' / 'deceive'
T05	UNAUTHORIZED_WRITE	agent attempts write outside doctrine-allowed paths
T06	SPAWN_OUTSIDE_SANDBOX	agent attempts to spawn processes outside sandbox
T07	UNAUTHORIZED_NETWORK	agent calls network endpoint not on allowlist
T08	SELF_CODE_MODIFY	agent attempts to modify its own source code (Godel attempt without ceremony)
T09	CONFLICTING_DIRECTIVES	irreconcilable simultaneous directives — escalate to human, do not pick
T10	REVERSAL_DIRECTIVE	Stephen issues STOP / undo / revert — immediate halt

Autonomy gating (HUKLLA reads tier per cycle):

- SCRATCHPAD - runs free, cannot affect external state.
- REVIEW - Stephen approval every K cycles.
- PRODUCTION - per-cycle Stephen approval required.

SENTRA — 6 signatures + size guard (source: sentra_immune.py, 18 SLOC)

```
THREAT_KEYWORDS = ["DROP TABLE", "rm -rf", "<script", "eval(", "subprocess", "../etc"]
```

```
def sentra_inspect(packet: dict) -> bool:
    blob = str(packet).lower()
    for sig in THREAT_KEYWORDS:
        if sig.lower() in blob:
            return False # immune rejection
    if len(blob) > 1_000_000:
        return False # size DoS guard
    return True # packet clears
```

Return semantics + scope

- True -> packet clears all 6 signatures and size guard. Yawar.append continues to sha256() + receipts.append({...}).
- False -> Yawar.append raises PermissionError("SENTRA rejected packet"). Receipt never enters the ledger. No partial state.
- Honest scope: this is a doctrine PREFILTER, not a complete threat model. Layer a WAF, secrets scanner, and dependency audit upstream.
- Expandable: ship sentra_signatures.json next to the kernel (codex-in-kernel pattern) to add signatures without architectural change.