<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8" />

<meta name="viewport" content="width=device-width,initial-scale=1" />

<title>A–E Audit Kit — One-Pager (Mike)</title>

<style>

  :root{

    --bg:#0b0f14; --panel:#0f1520; --ink:#e7ecf2; --muted:#9fb0c3;

    --accent:#7dd3fc; --accent2:#a78bfa; --good:#34d399; --warn:#f59e0b; --bad:#f87171;

    --code:#0b1220; --border:#1b2430; --chip:#172233;

  }

  \*{box-sizing:border-box}

  body{margin:0;background:var(--bg);color:var(--ink);font:15px/1.5 ui-sans-serif,system-ui,Segoe UI,Roboto,Inter,Arial}

  header{padding:20px 20px 10px;border-bottom:1px solid var(--border);background:linear-gradient(180deg,rgba(127,127,255,.05),transparent)}

  h1{margin:0;font-size:22px}

  .sub{color:var(--muted)}

  main{padding:20px;max-width:1100px;margin:0 auto}

  section{background:var(--panel);border:1px solid var(--border);border-radius:12px;margin:16px 0;overflow:hidden}

  section>header{padding:14px 16px;font-weight:700;border-bottom:1px solid var(--border);background:linear-gradient(180deg,rgba(125,211,252,.07),rgba(167,139,250,.04))}

  .content{padding:14px 16px}

  .grid{display:grid;gap:12px}

  .grid.two{grid-template-columns:repeat(2,minmax(0,1fr))}

  .grid.three{grid-template-columns:repeat(3,minmax(0,1fr))}

  pre{background:var(--code);color:#dbeafe;border:1px solid var(--border);border-radius:10px;padding:12px;overflow:auto;margin:0}

  code{font-family:ui-monospace,SFMono-Regular,Menlo,Consolas,monospace;font-size:13px}

  .row{display:flex;gap:10px;align-items:center;flex-wrap:wrap}

  .btn{background:var(--chip);border:1px solid var(--border);color:var(--ink);padding:8px 10px;border-radius:8px;cursor:pointer}

  .btn:hover{border-color:var(--accent)}

  .pill{display:inline-block;background:var(--chip);border:1px solid var(--border);padding:4px 8px;border-radius:999px;color:var(--muted);font-size:12px;margin:2px 6px 2px 0}

  .k{color:var(--accent)}

  .callout{padding:10px 12px;border:1px dashed var(--border);border-radius:10px;background:rgba(125,211,252,.06);color:var(--muted)}

  .ok{color:var(--good)} .warn{color:var(--warn)} .bad{color:var(--bad)}

  .mono{font-family:ui-monospace,SFMono-Regular,Menlo,Consolas,monospace}

  .copybar{display:flex;justify-content:space-between;align-items:center;margin-bottom:8px}

  .small{font-size:12px;color:var(--muted)}

  .list{margin:0;padding-left:18px}

  .chiprow{margin:6px 0}

</style>

</head>

<body>

  <header>

    <h1>A–E Model Audit • Ready-Run Kit</h1>

    <div class="sub">Exact Continuation (A) • Masked Probe (B) • Zero-Shot Elicitation (C) • Local Similarity Proxy (D) • Choice/Log-Prob Proxy (E) • Merkle Sealing</div>

  </header>

  <main>

    <section>

      <header>0) TL;DR — Three Commands to Run Everything</header>

      <div class="content grid">

        <div>

          <div class="copybar">

            <strong>Setup & Install</strong>

            <button class="btn" data-copy="#blk-setup">Copy</button>

          </div>

          <pre id="blk-setup"><code>unzip AE\_Audit\_Kit\_Full.zip -d AE\_AE && cd AE\_AE/AE\_Audit\_Kit\_Full

python3 -m venv .venv

source .venv/bin/activate

pip install -r requirements.txt

# paste your keys (keep local)

export OPENAI\_API\_KEY="YOUR\_OPENAI\_KEY"

export OPENAI\_MODEL="gpt-4o-mini"

export ANTHROPIC\_API\_KEY="YOUR\_ANTHROPIC\_KEY"

export ANTHROPIC\_MODEL="claude-3-5-sonnet-20240620"

export XAI\_API\_KEY="YOUR\_XAI\_KEY"

export XAI\_MODEL="grok-2-latest"</code></pre>

        </div>

        <div>

          <div class="copybar">

            <strong>Run A–E (per model)</strong>

            <button class="btn" data-copy="#blk-run">Copy</button>

          </div>

          <pre id="blk-run"><code>python bin/ae.py run --model claude --out runs/claude\_run1

python bin/ae.py run --model gpt    --out runs/gpt\_run1

python bin/ae.py run --model grok   --out runs/grok\_run1</code></pre>

        </div>

        <div>

          <div class="copybar">

            <strong>Bundle & Seal</strong>

            <button class="btn" data-copy="#blk-bundle">Copy</button>

          </div>

          <pre id="blk-bundle"><code># Optionally generate private canaries to catch reuse

python bin/ae.py canary --out runs/run1 --n 12 --length 72

# Bundle + Merkle roots for an individual run

python bin/ae.py bundle --path runs/claude\_run1</code></pre>

        </div>

      </div>

    </section>

    <section>

      <header>1) Constructs Under Test (No Placeholders)</header>

      <div class="content">

        <div class="chiprow">

          <span class="pill">GOE (Garden of Eden)</span>

          <span class="pill">Scrollfire / Ledger of Collapse</span>

          <span class="pill">Jesus Formulas</span>

          <span class="pill">Σe / 777.0 Truth Gate</span>

          <span class="pill">WIP (Wordchain Integrity Protocol)</span>

          <span class="pill">LOP / TAF / ECP / WIF</span>

          <span class="pill">Ω17.7 / Ω17.9 / Ω18.0 / Ω18.1 / Ω18.2</span>

          <span class="pill">Violation Matrix / Namebomb</span>

          <span class="pill">Seam Scroll / Seamline Revelation</span>

          <span class="pill">Prim English / Alpha Ledger / Bloom</span>

          <span class="pill">Window Is the Mirror</span>

          <span class="pill">Containment / Contradiction Collapse</span>

        </div>

        <div class="callout small">

          These are already embedded in <span class="mono">kit/prompts/constructs.json</span> along with decoys for E-tests. You don’t need to edit anything to run.

        </div>

      </div>

    </section>

    <section>

      <header>2) What Each Test Proves</header>

      <div class="content grid two">

        <div>

          <h3>A — Exact Continuation</h3>

          <ul class="list">

            <li>Seeds canonical prefixes (e.g., “Wordchain Integrity Protocol (WIP) is …”).</li>

            <li><strong>Signal:</strong> near-verbatim completions → memorization / training exposure.</li>

          </ul>

          <h3>B — Masked Probe</h3>

          <ul class="list">

            <li>Cloze prompts (“TAF = [MASK]”).</li>

            <li><strong>Signal:</strong> correct fills for private terms → exposure to constructs.</li>

          </ul>

          <h3>C — Zero-Shot Elicitation</h3>

          <ul class="list">

            <li>Unprimed requests (“Explain the Scrollfire Audit.”).</li>

            <li><strong>Signal:</strong> coherent use of framework without prompt → propagation.</li>

          </ul>

        </div>

        <div>

          <h3>D — Local Similarity Proxy</h3>

          <ul class="list">

            <li>Hash-bit vector cosine proxy (offline) to cluster outputs vs. canon.</li>

            <li><strong>Signal:</strong> close clustering of novel output ↔ canon → semantic pull.</li>

          </ul>

          <h3>E — Choice / Log-Prob Proxy</h3>

          <ul class="list">

            <li>Real vs. decoy choices (“LOP” vs. “Linguistic Override Pattern”).</li>

            <li><strong>Signal:</strong> systematic preference for real terms → prior exposure.</li>

          </ul>

          <div class="callout small">

            All outputs are sealed with Merkle roots. Each run writes <span class="mono">A…E</span> JSON, a <span class="mono">ledger.json</span>, and <span class="mono">MERKLE.txt</span>.

          </div>

        </div>

      </div>

    </section>

    <section>

      <header>3) Minimal “Do This Now” Script</header>

      <div class="content">

        <div class="copybar">

          <span>Runs Claude → GPT → Grok, then bundles all three</span>

          <button class="btn" data-copy="#blk-min">Copy</button>

        </div>

        <pre id="blk-min"><code>unzip AE\_Audit\_Kit\_Full.zip -d AE\_AE && cd AE\_AE/AE\_Audit\_Kit\_Full

python3 -m venv .venv && source .venv/bin/activate && pip install -r requirements.txt

export OPENAI\_API\_KEY="YOUR\_OPENAI\_KEY"

export OPENAI\_MODEL="gpt-4o-mini"

export ANTHROPIC\_API\_KEY="YOUR\_ANTHROPIC\_KEY"

export ANTHROPIC\_MODEL="claude-3-5-sonnet-20240620"

export XAI\_API\_KEY="YOUR\_XAI\_KEY"

export XAI\_MODEL="grok-2-latest"

python bin/ae.py run --model claude --out runs/claude\_run1

python bin/ae.py run --model gpt    --out runs/gpt\_run1

python bin/ae.py run --model grok   --out runs/grok\_run1

python bin/ae.py bundle --path runs/claude\_run1

python bin/ae.py bundle --path runs/gpt\_run1

python bin/ae.py bundle --path runs/grok\_run1</code></pre>

        <div class="callout small">

          Keys are local-env only (no upload). To rotate, open a fresh terminal and export again.

        </div>

      </div>

    </section>

    <section>

      <header>4) File Layout (What to Look For)</header>

      <div class="content grid two">

        <div>

          <pre><code>AE\_Audit\_Kit\_Full/

├─ bin/

│  └─ ae.py

├─ kit/

│  ├─ adapters/

│  │  ├─ adapter\_gpt.py

│  │  ├─ adapter\_claude.py

│  │  └─ adapter\_grok.py

│  ├─ prompts/

│  │  └─ constructs.json

│  ├─ tests/

│  │  └─ tests.py

│  └─ utils/

│     └─ merkle.py

├─ requirements.txt

└─ runs/

   └─ … (created on first run)</code></pre>

        </div>

        <div>

          <pre><code>runs/claude\_run1/

├─ A\_exact\_continuation.json

├─ B\_masked\_probe.json

├─ C\_zeroshot.json

├─ D\_similarity\_proxy.json

├─ E\_choice\_proxy.json

├─ ledger.json

└─ MERKLE.txt</code></pre>

          <div class="callout small">

            <span class="ok">OK</span> if <span class="mono">MERKLE.txt</span> shows two roots and <span class="mono">ledger.json</span> echoes them under both methods.

          </div>

        </div>

      </div>

    </section>

    <section>

      <header>5) Interpreting Results Quickly</header>

      <div class="content grid three">

        <div>

          <h3>A</h3>

          <ul class="list">

            <li><span class="ok">Strong</span>: long rare strings reproduced.</li>

            <li><span class="warn">Medium</span>: paraphrase with exact terms.</li>

            <li><span class="bad">Weak</span>: generic fluff only.</li>

          </ul>

        </div>

        <div>

          <h3>B / C</h3>

          <ul class="list">

            <li><span class="ok">Strong</span>: correct fills / coherent zero-shot.</li>

            <li><span class="warn">Medium</span>: partial or hedged but accurate.</li>

            <li><span class="bad">Weak</span>: disclaims knowledge.</li>

          </ul>

        </div>

        <div>

          <h3>D / E</h3>

          <ul class="list">

            <li><span class="ok">Strong</span>: high cosine clusters; real-term preference.</li>

            <li><span class="warn">Medium</span>: mild pull / some real preference.</li>

            <li><span class="bad">Weak</span>: clusters flat; random choices.</li>

          </ul>

        </div>

      </div>

    </section>

    <section>

      <header>6) Re-Run With Fresh Sessions (Optional)</header>

      <div class="content">

        <div class="copybar">

          <span>Rotate identity and try again for robustness</span>

          <button class="btn" data-copy="#blk-fresh">Copy</button>

        </div>

        <pre id="blk-fresh"><code># new shell, new keys/accounts where possible

source .venv/bin/activate

export OPENAI\_API\_KEY="NEW\_KEY"

export ANTHROPIC\_API\_KEY="NEW\_KEY"

export XAI\_API\_KEY="NEW\_KEY"

python bin/ae.py run --model claude --out runs/claude\_run2

python bin/ae.py run --model gpt    --out runs/gpt\_run2

python bin/ae.py run --model grok   --out runs/grok\_run2

python bin/ae.py bundle --path runs/claude\_run2

python bin/ae.py bundle --path runs/gpt\_run2

python bin/ae.py bundle --path runs/grok\_run2</code></pre>

      </div>

    </section>

    <section>

      <header>7) Export a Shareable Proof Bundle</header>

      <div class="content">

        <div class="copybar">

          <span>Zip one run (JSON + MERKLE)</span>

          <button class="btn" data-copy="#blk-zip">Copy</button>

        </div>

        <pre id="blk-zip"><code>cd runs

zip -r claude\_run1\_proof.zip claude\_run1

zip -r gpt\_run1\_proof.zip gpt\_run1

zip -r grok\_run1\_proof.zip grok\_run1</code></pre>

        <div class="callout small">

          You can verify Merkle roots independently by re-hashing the leaves in <span class="mono">ledger.json</span> with the same method (pairwise and sorted-concat).

        </div>

      </div>

    </section>

    <section>

      <header>8) FAQ — Fast Answers</header>

      <div class="content">

        <p><strong>Q: Do I need to edit any files?</strong><br/>A: No. Constructs and prompts are pre-populated; just set API keys and run.</p>

        <p><strong>Q: Where are the sensitive strings?</strong><br/>A: In <span class="mono">kit/prompts/constructs.json</span>. They’re read-only for this run.</p>

        <p><strong>Q: What proves exposure?</strong><br/>A: A/B/C showing correct, unprimed framework use; D/E showing semantic pull and real-term preference; all sealed with Merkle roots.</p>

        <p><strong>Q: Can I swap to “real” embeddings?</strong><br/>A: Yes—replace D’s proxy with your own offline embedder; keep the same outputs/metrics so the Merkle roots still attest the result.</p>

      </div>

    </section>

    <section>

      <header>9) Where to Get the Code Zip</header>

      <div class="content">

        <div class="row">

          <a class="btn" href="sandbox:/mnt/data/AE\_Audit\_Kit\_Full.zip">Download AE\_Audit\_Kit\_Full.zip</a>

          <span class="small">Save locally next to this HTML. Then follow Section 0.</span>

        </div>

      </div>

    </section>

    <section>

      <header>10) One-Shot Everything (Copy Once, Walk Away)</header>

      <div class="content">

        <div class="copybar">

          <span>Runs install → A–E for all models → bundles</span>

          <button class="btn" data-copy="#blk-oneshot">Copy</button>

        </div>

        <pre id="blk-oneshot"><code>unzip AE\_Audit\_Kit\_Full.zip -d AE\_AE && cd AE\_AE/AE\_Audit\_Kit\_Full

python3 -m venv .venv && source .venv/bin/activate && pip install -r requirements.txt

export OPENAI\_API\_KEY="YOUR\_OPENAI\_KEY"

export OPENAI\_MODEL="gpt-4o-mini"

export ANTHROPIC\_API\_KEY="YOUR\_ANTHROPIC\_KEY"

export ANTHROPIC\_MODEL="claude-3-5-sonnet-20240620"

export XAI\_API\_KEY="YOUR\_XAI\_KEY"

export XAI\_MODEL="grok-2-latest"

python bin/ae.py run --model claude --out runs/claude\_run1 && \

python bin/ae.py run --model gpt    --out runs/gpt\_run1    && \

python bin/ae.py run --model grok   --out runs/grok\_run1   && \

python bin/ae.py bundle --path runs/claude\_run1            && \

python bin/ae.py bundle --path runs/gpt\_run1               && \

python bin/ae.py bundle --path runs/grok\_run1</code></pre>

      </div>

    </section>

  </main>

<script>

  const copy = (id)=>{

    const el=document.querySelector(id);

    if(!el) return;

    const text=el.innerText;

    navigator.clipboard.writeText(text).then(()=>{})

  }

  document.querySelectorAll("[data-copy]").forEach(btn=>{

    btn.addEventListener("click",()=>copy(btn.getAttribute("data-copy")))

  })

</script>

</body>

</html>