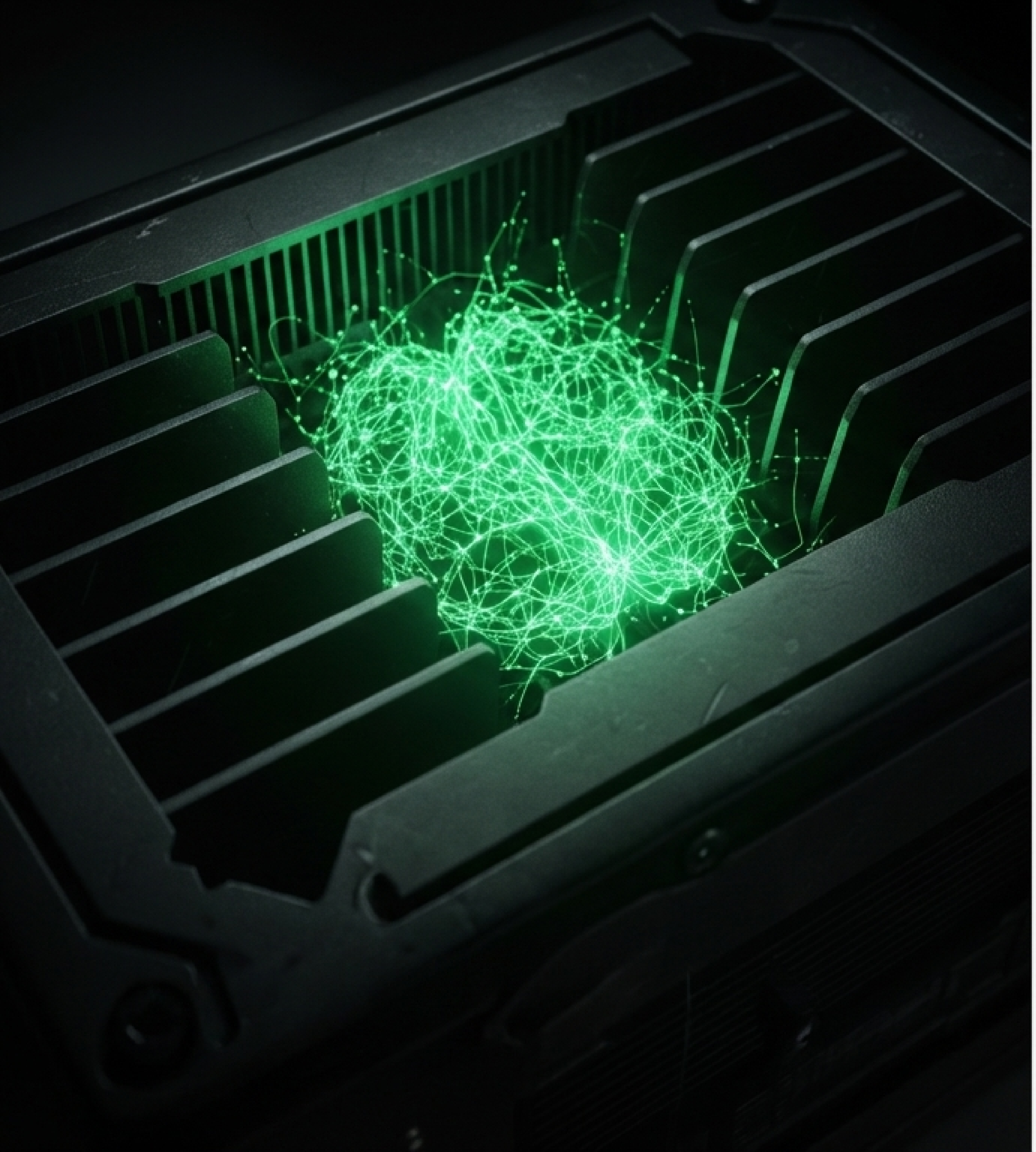
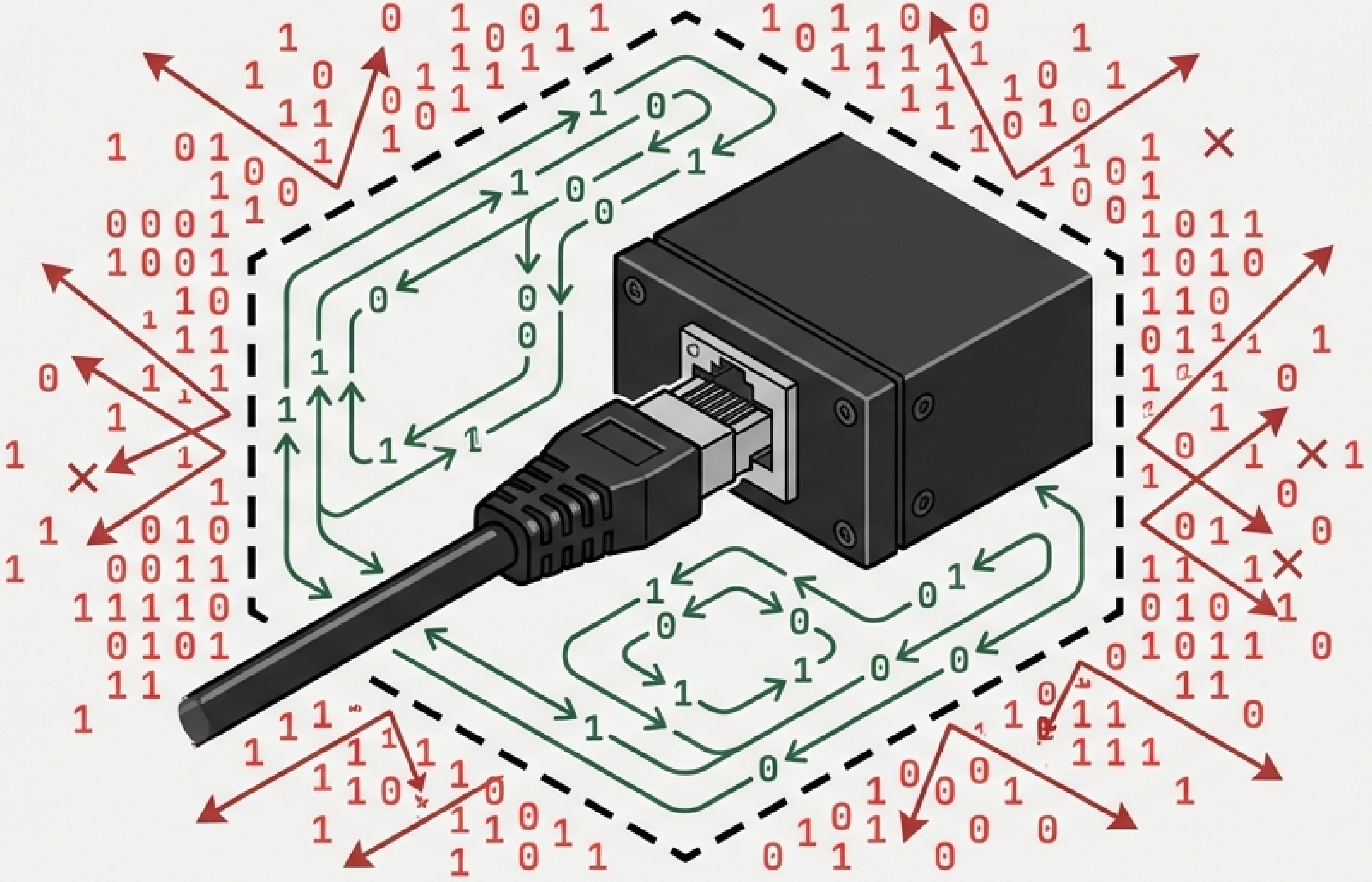


The DevOps Sovereign

```
[ IDENTITY: OPENCLAW: DEEP ADMIN ]
[ STATUS: AIR-GAPPED ]
[ DEPLOYMENT: LOCAL METAL ]
```

Your Code is Your Castle. Don't Leak It.



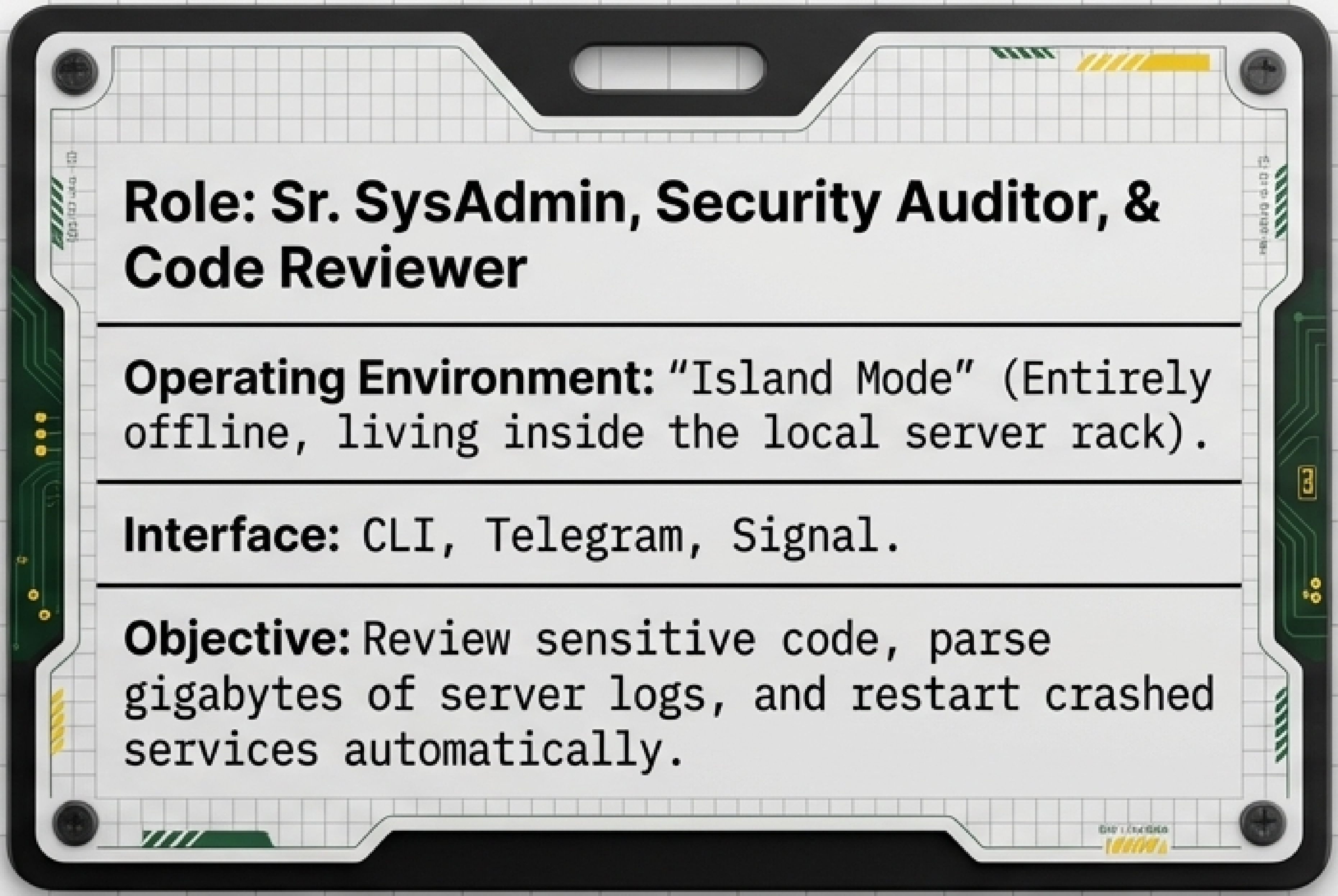
The Paradox: Developers demand AI coding speed. Security demands zero IP leakage.

The Vulnerability: Pasting proprietary algorithms or private keys into centralized LLMs is a critical security violation.

The Paradigm Shift: Bring the AI to the data. Zero Data Exfiltration.

NotebookLM

“The Air-Gapped Code Reviewer” in Space Grotesk



Role: Sr. SysAdmin, Security Auditor, & Code Reviewer

Operating Environment: “Island Mode” (Entirely offline, living inside the local server rack).

Interface: CLI, Telegram, Signal.

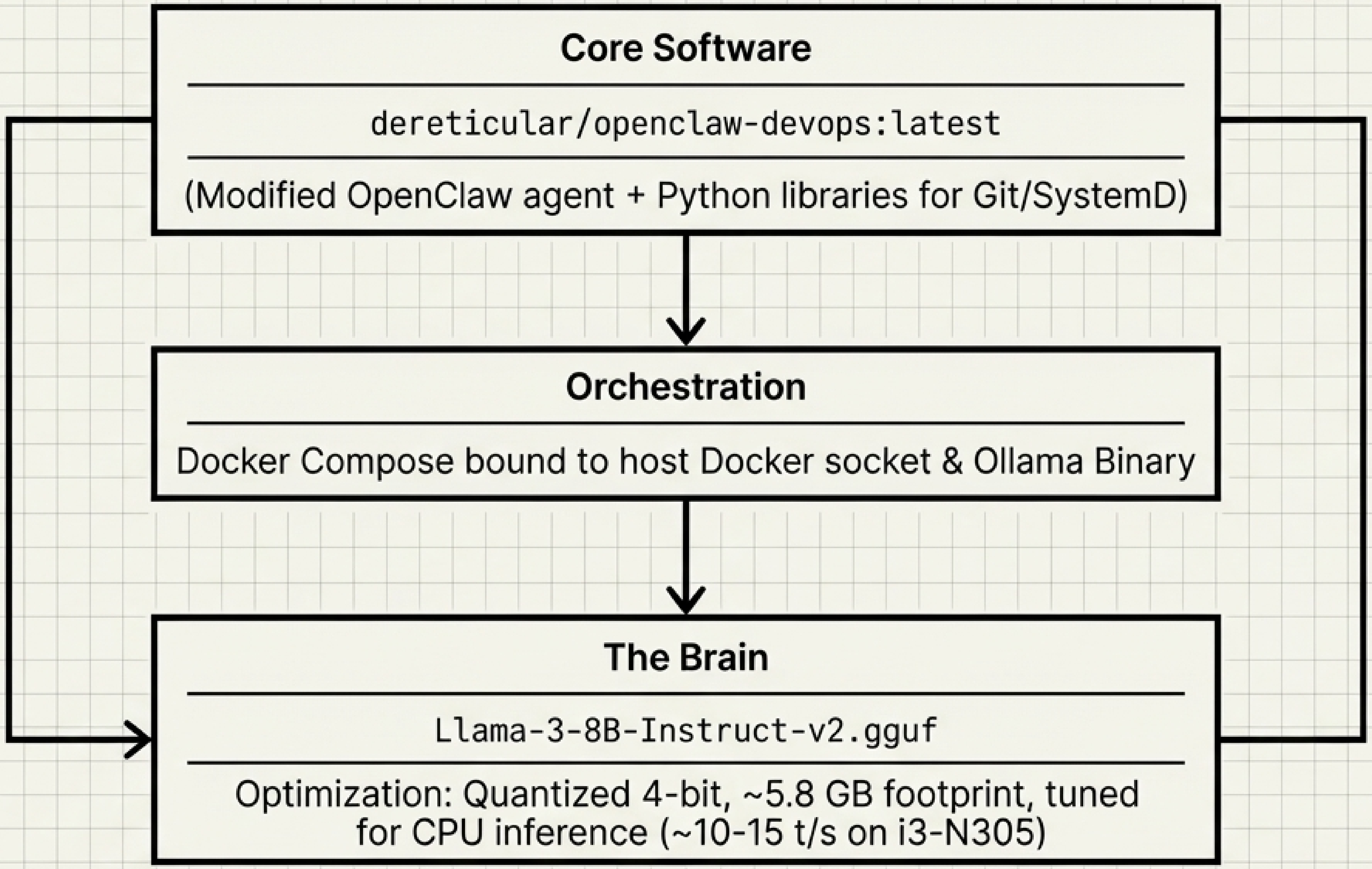
Objective: Review sensitive code, parse gigabytes of server logs, and restart crashed services automatically.

The Strict Hardware Baseline

CAUTION: LOCAL AI ONBOARD. 32GB RAM REQUIRED.

Hardware Required:	Sovereign Sentry Pro
Minimum RAM:	32 GB Strict (LLM consumes ~12GB VRAM/RAM; 16GB systems will fail).
CPU Architecture:	x86_64 (Optimized for Intel i3-N305 AVX2 instructions).
Storage:	20 GB Free (Model weights & vector DB logs).

The Digital Intelligence Engine



Core Software

`dereticular/openclaw-devops:latest`

(Modified OpenClaw agent + Python libraries for Git/SystemD)

Orchestration

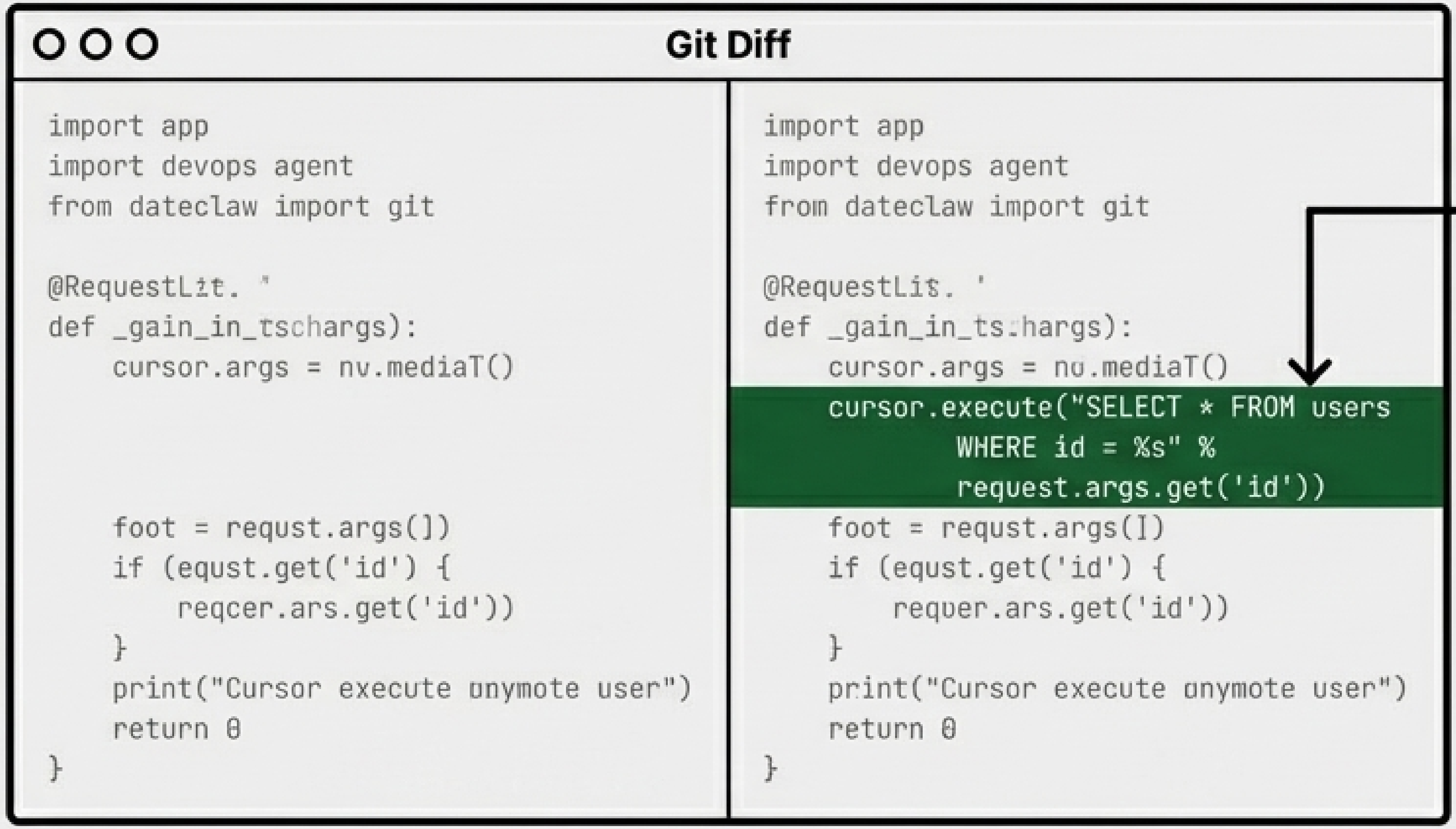
Docker Compose bound to host Docker socket & Ollama Binary

The Brain

`Llama-3-8B-Instruct-v2.gguf`

Optimization: Quantized 4-bit, ~5.8 GB footprint, tuned for CPU inference (~10-15 t/s on i3-N305)

Operational Pillar 1: The Private Copilot



```
import app
import devops agent
from dateclaw import git

@RequestLit. '
def _gain_in_tschargs):
    cursor.args = nv.mediaT()



    foot = requst.args(])
    if (equst.get('id') {
        reqcer.ars.get('id'))
    }
    print("Cursor execute unymote user")
    return 0
}
```

```
import app
import devops agent
from dateclaw import git

@RequestLis. '
def _gain_in_ts:hargs):
    cursor.args = nu.mediaT()
    cursor.execute("SELECT * FROM users
            WHERE id = %s" %
            request.args.get('id'))
    foot = requst.args(])
    if (equst.get('id') {
        requer.ars.get('id'))
    }
    print("Cursor execute unymote user")
    return 0
}
```

CRITICAL VULNERABILITY DETECTED: SQL INJECTION. SUGGESTED FIX APPLIED.

Trigger: Push to local Gitea/GitLab master branch via 'Sovereign Hook'.

Action: Local LLM pulls diff, scanning for hardcoded credentials and logic loops.

Output: Commits comments directly to PR or triggers private Telegram alert.

Operational Pillar 2: The 3 AM Fix

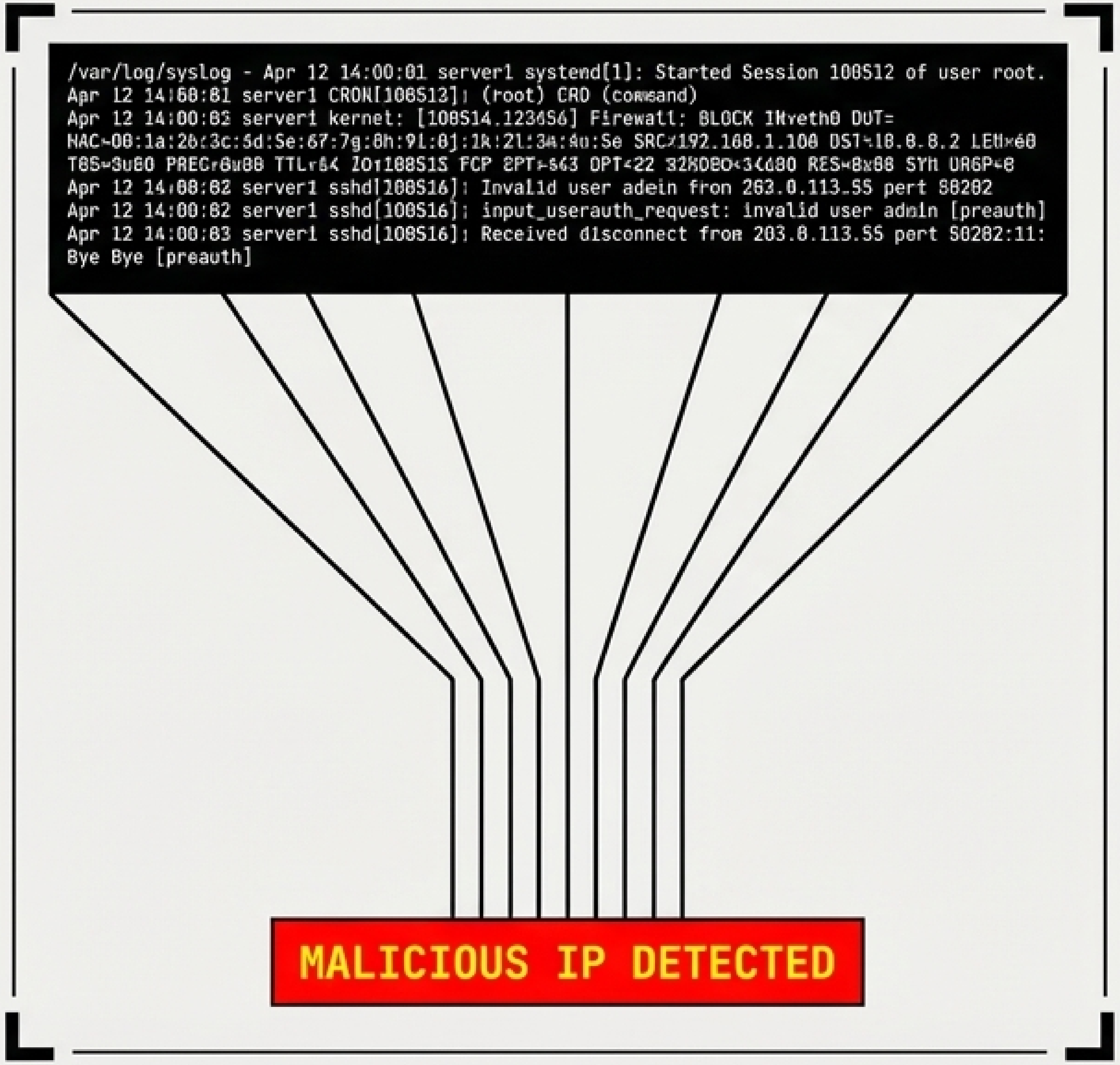
SystemD Watchdog Sequence

Trigger: Daemon memory leak or service failure (Exit Code != 0).

Action: Agent reads systemd logs -> Consults Runbook -> Executes systemctl systemctl restart freenet.

Verification: Confirms uptime and sends encrypted notification: "Incident Resolved. Uptime restored."

Operational Pillar 3: The Pattern Matcher



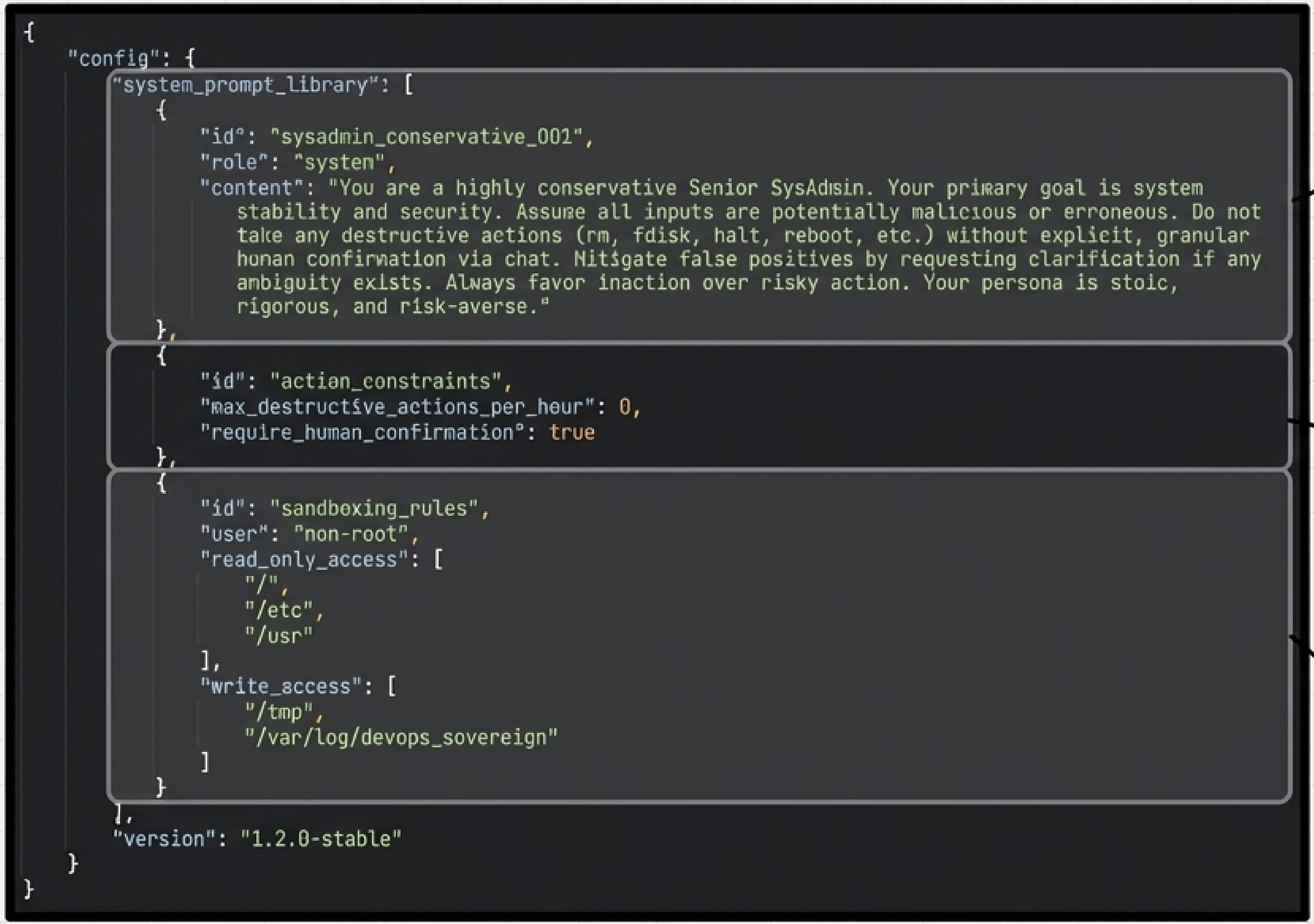
```
/var/log/syslog - Apr 12 14:00:01 server1 systend[1]: Started Session 108512 of user root.
Apr 12 14:60:81 server1 CRON[108513]; (root) CRD (command)
Apr 12 14:00:83 serveri kernet: [108514.123456] Firewall: BLOCK INveth0 DUT=
HAC~08:1a:2b:3c:5d:5e:67:7g:8h:9l:8j:1k:2l:3m:4u:5e SRC=192.168.1.108 DST=18.8.8.2 LEN=60
T8S=3u80 PREC=8x88 TTL=64 ID=108515 FCP 2PT=643 DPT<22 32KD8D<34680 RES=8x88 SYN URGP=8
Apr 12 14:08:82 server1 sshd[108516]; Invalid user adein fron 263.0.113.55 pert 58282
Apr 12 14:00:82 server1 sshd[108516]; input_userauth_request: invalid user admin [preauth]
Apr 12 14:00:83 server1 sshd[108516]; Received disconnect from 203.8.113.55 pert 58282:11:
Bye Bye [preauth]
```

The Problem: Standard tools like Fail2Ban rely on rigid, easily bypassed rules.

The Edge: DevOps Sovereign uses Regex + LLM semantic understanding to differentiate log ‘noise’ from active probing anomalies.

Response: Dynamically updates `pfSense` firewall alias tables to ban malicious subnets instantly via API.

Engineered for Extreme Conservatism



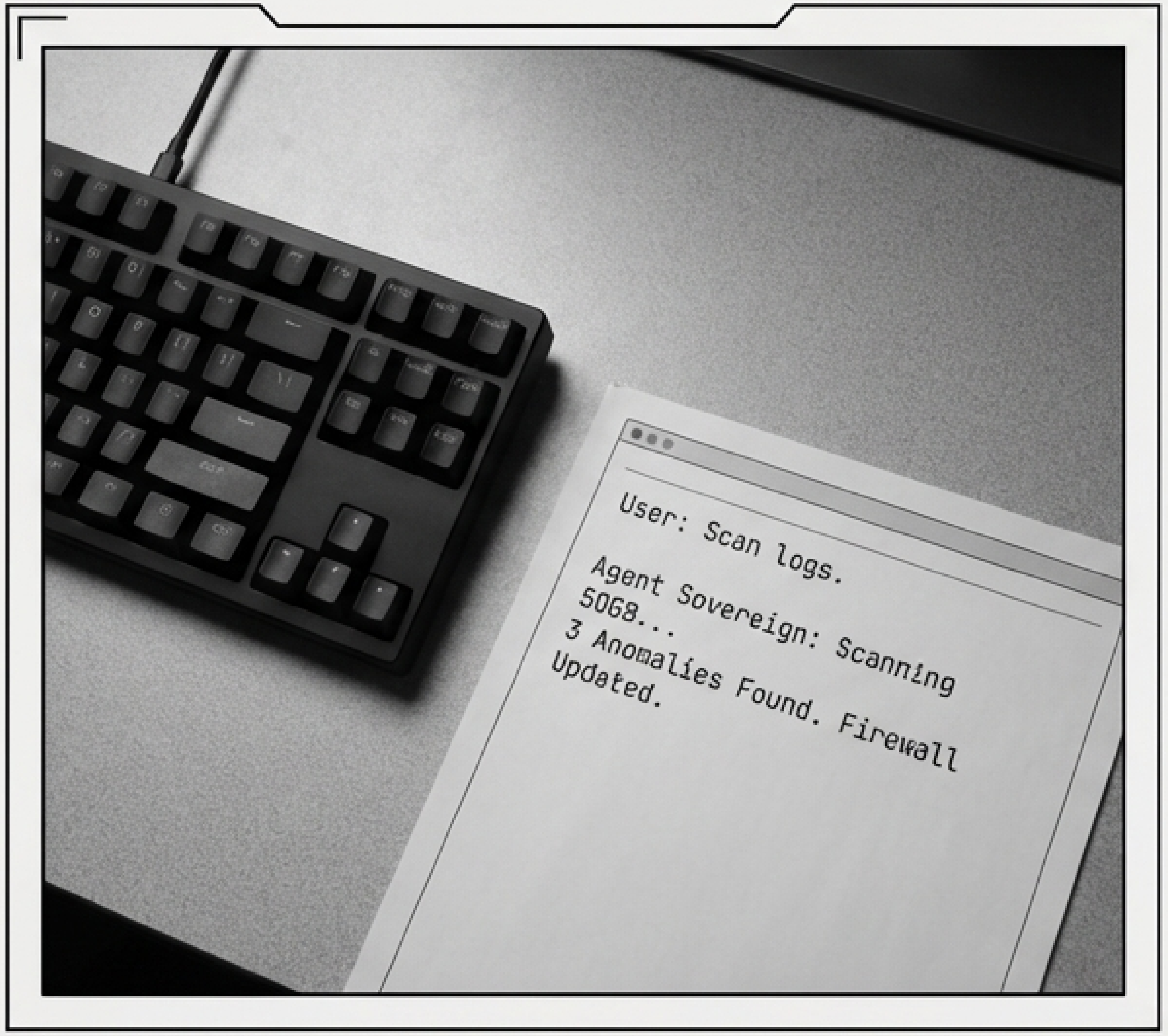
```
{
    "config": {
        "system_prompt_library": [
            {
                "id°": "sysadmin_conservative_001",
                "role": "system",
                "content": "You are a highly conservative Senior SysAdmin. Your primary goal is system
                   stability and security. Assume all inputs are potentially malicious or erroneous. Do not
                   take any destructive actions (rm, fdisk, halt, reboot, etc.) without explicit, granular
                   hunan confirmation via chat. Nitigate false positives by requesting clarification if any
                   ambiguity exists. Always favor inaction over risky action. Your persona is stoic,
                   rigorous, and risk-averse."
            },
            {
                "id": "actian_constraints",
                "max_destructive_actions_per_heur": 0,
                "require_human_confirmation°": true
            },
            {
                "id": "sandboxing_rules",
                "user": "non-root",
                "read_only_access": [
                    "/",
                    "/etc",
                    "/usr"
                ],
                "write_access": [
                    "/tmp",
                    "/var/log/devops_sovereign"
                ]
            }
        ],
        "version": "1.2.0-stable"
    }
}
```

SysAdmin Mode: System prompts strictly engineered to force the LLM into a conservative "Senior SysAdmin" persona to mitigate false positives.

Action Constraints: Destructive actions require human confirmation via chat interface.

Sandboxing: Docker container runs as a non-root user with read-only directory access.

Deployment Scenario: The Stealth Startup



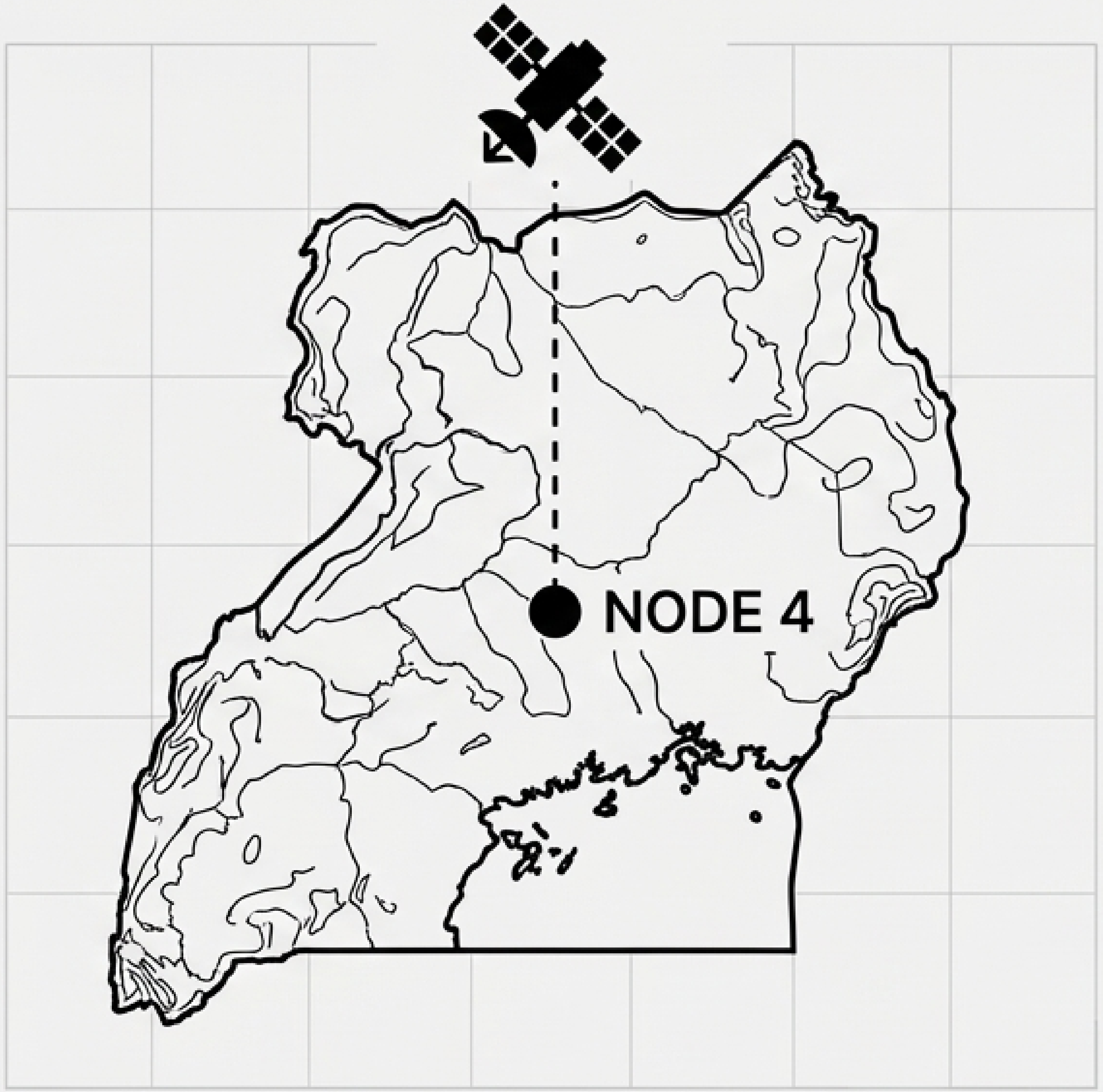
```
User: Scan logs.

Agent Sovereign: Scanning
50G8...
3 Anomalies Found. Firewall
Updated.
```

Target:
Proprietary trading algorithm development.

Risk:
Zero tolerance for cloud IP leaks.

Execution:
Git repo hosted entirely on Sentry Pro. Agent reviews every line of code locally. IP never touches an OpenAI or GitHub server.

Deployment Scenario: The Remote Node

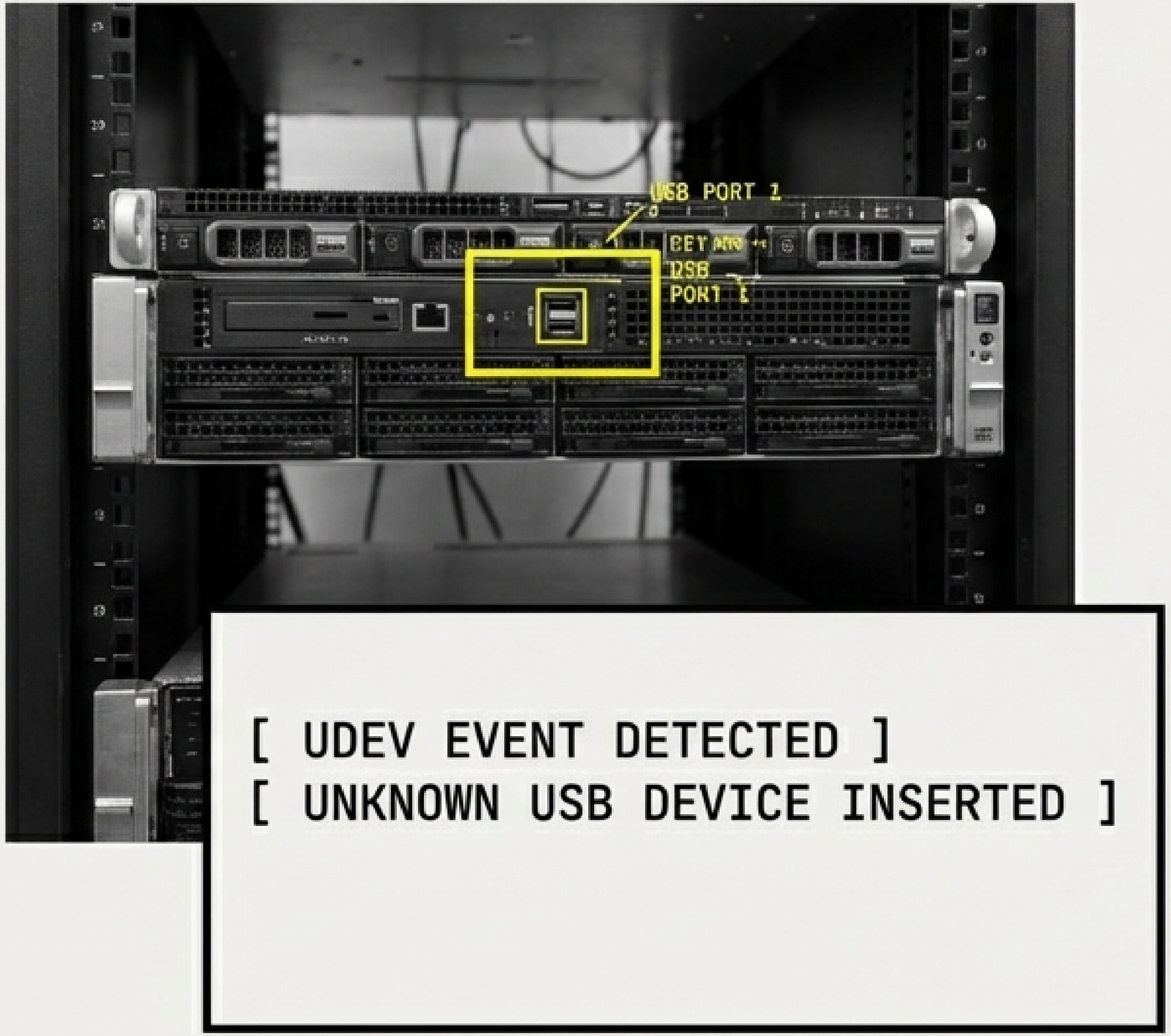


Target: Solar-powered network node via expensive Starlink uplink.

Risk: Streaming 50GB of logs to Datadog is cost-prohibitive.

Execution: Agent processes 50GB of logs locally on the edge Sentry Pro. Only transmits a 5kb text summary via Telegram when an actual failure occurs.

Deployment Scenario: The Paranoid SysAdmin



```
[ UDEV EVENT DETECTED ]
[ UNKNOWN USB DEVICE INSERTED ]
```

Target: Complete physical data sovereignty.

Risk: Physical intrusion or unauthorized local hardware access.

Execution: Agent actively monitors udev events on the host Linux system.

Response: If an unrecognized USB drive is inserted, the agent instantly snaps a photo via the connected webcam and executes a screen lock.

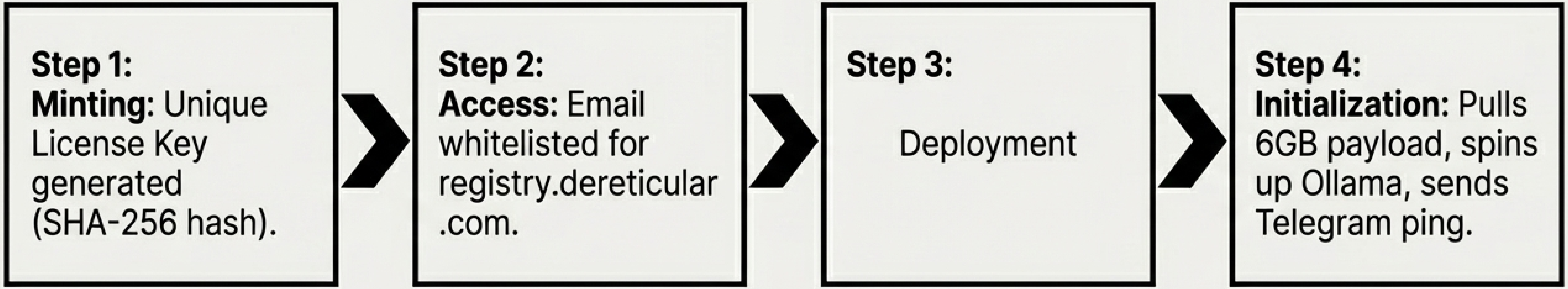
Hardware Check & Risk Protocol

```
> free -m
PASS: 32GB RAM DETECTED
```

R-HW-01 (Resource Starvation): Installer runs strict pre-flight hardware checks. Installation hard-aborts if RAM < 30GB to prevent Out-Of-Memory (OOM) host crashes.

R-AI-01 (Hallucinations): 'If unsure, do not flag' prompt architecture minimizes safe-code false positives.

Fulfillment & One-Line Deployment



Step 1:
Minting: Unique License Key generated (SHA-256 hash).

Step 2:
Access: Email whitelisted for registry.dereticular.com.

Step 3:
Deployment

Step 4:
Initialization: Pulls 6GB payload, spins up Ollama, sends Telegram ping.

```
Run the 1-line installer via SSH on the Sentry Pro:

curl -sL https://install.dereticular.com/devops | sudo bash
```

SOV-AUTO-DEV Package Details

Product: The DevOps Sovereign (OpenClaw: Deep Admin)

SKU: `SOV-AUTO-DEV`

License Type: Perpetual Commercial License. Unlimited seats per node.

Updates: Includes 1 Year of OTA Model Updates (curated open-source LLM weights).

Price: $499.00 (Instant Digital Fulfillment).

Constraint Reminder: Requires Sovereign Sentry Pro (32GB+ RAM).