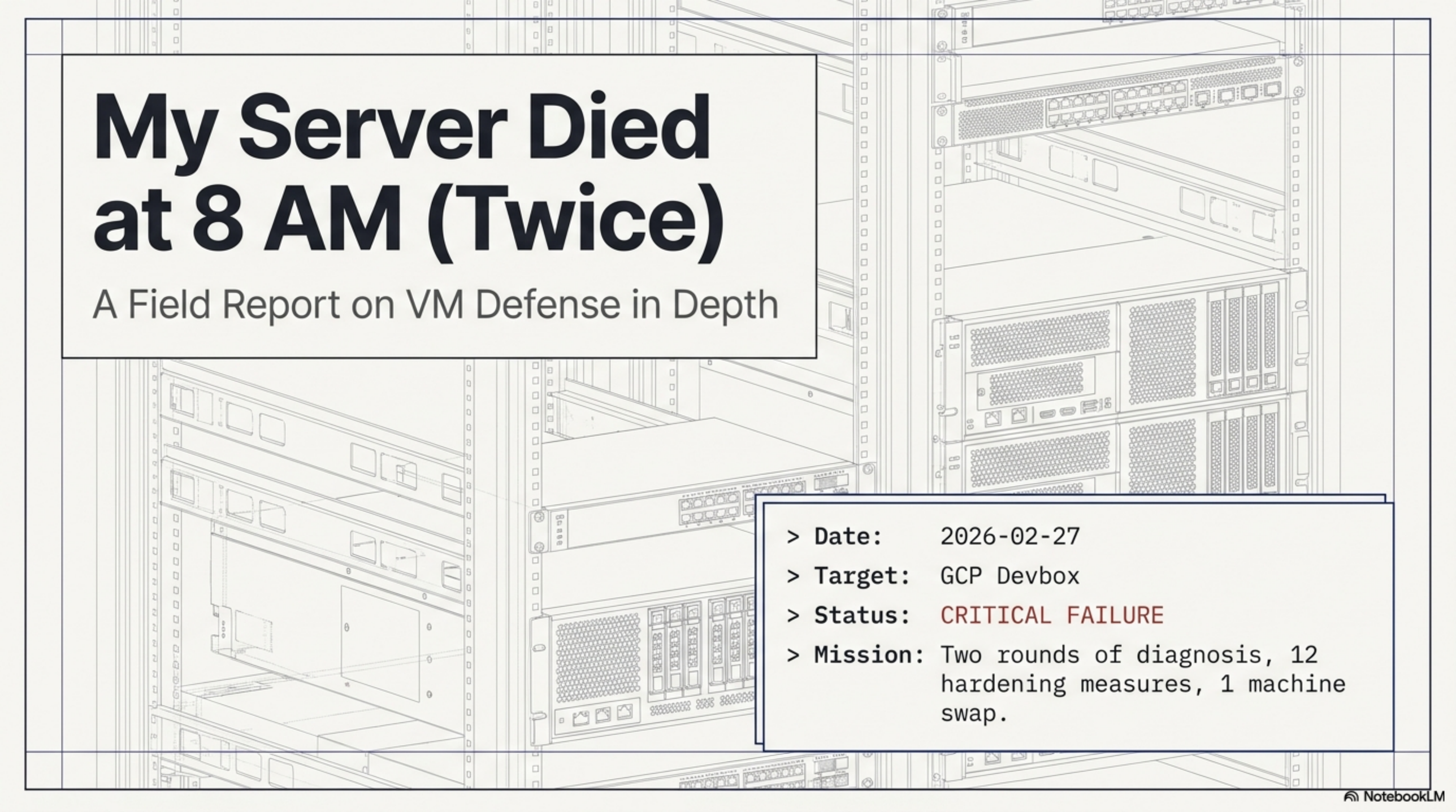
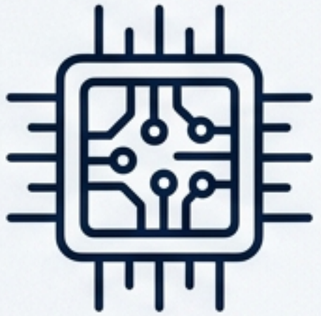


My Server Died at 8 AM (Twice)

A Field Report on VM Defense in Depth

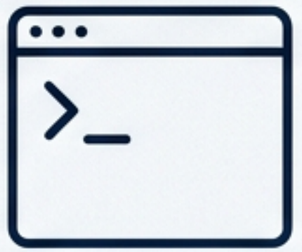
> **Date:** 2026-02-27
> **Target:** GCP Devbox
> **Status:** CRITICAL FAILURE
> **Mission:** Two rounds of diagnosis, 12 hardening measures, 1 machine swap.

The Baseline Configuration



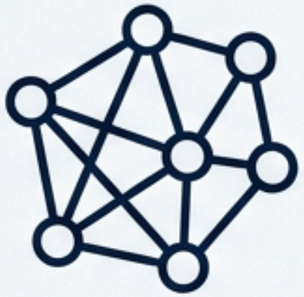
Hardware

GCP c3-standard-4 | 4 vCPU | 16GB RAM | $152/month



Operating System

Ubuntu 24.04



Workload

Long-lived services | Headless Chrome managed by PM2 | Dockerized AI API gateways

Status

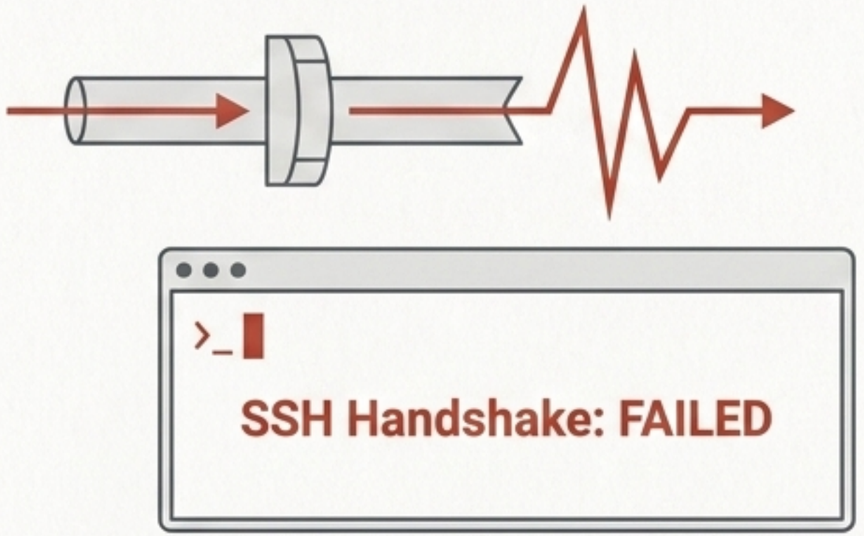
Last Verified: 06:00 AM.
System stable.
Uptime: 33 hours.

8:00 AM: The First Death

The Network Layer

Port 22 is open. The kernel networking stack is alive and responding.

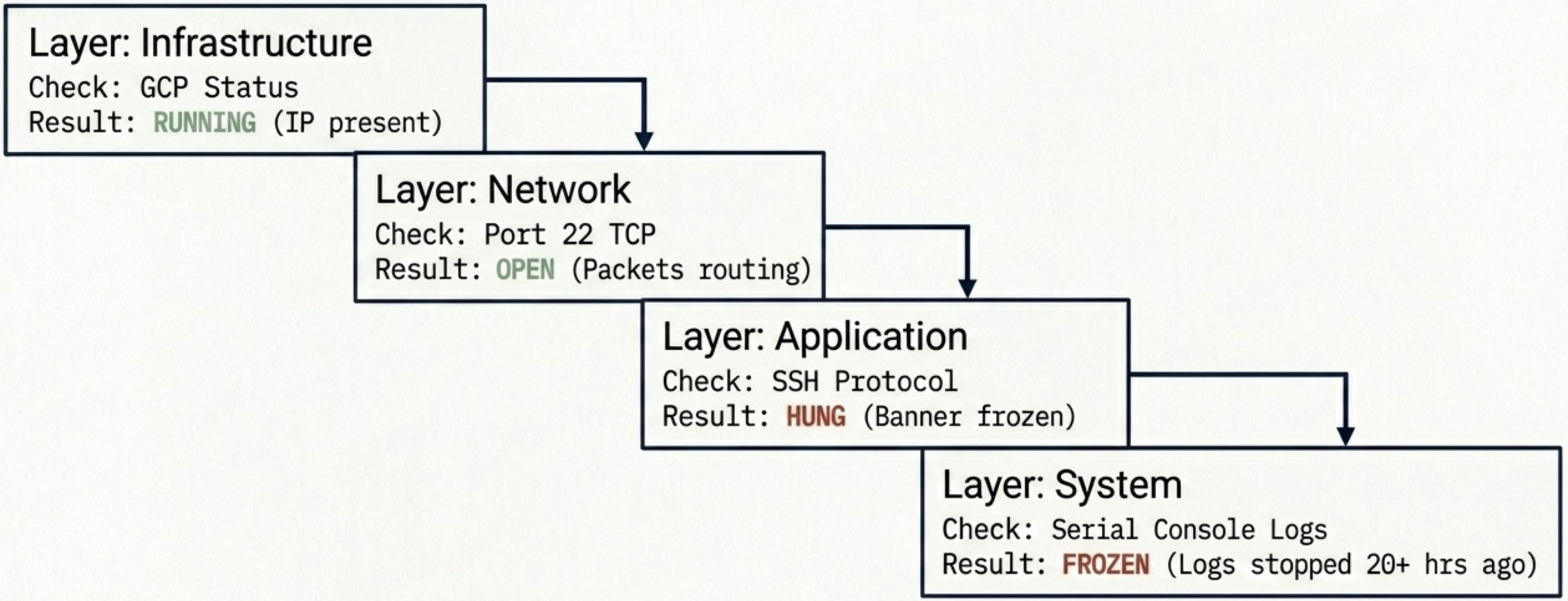
The Application Layer



Connection hangs entirely at “banner exchange” (Step 2 of SSH). The userspace is completely frozen.

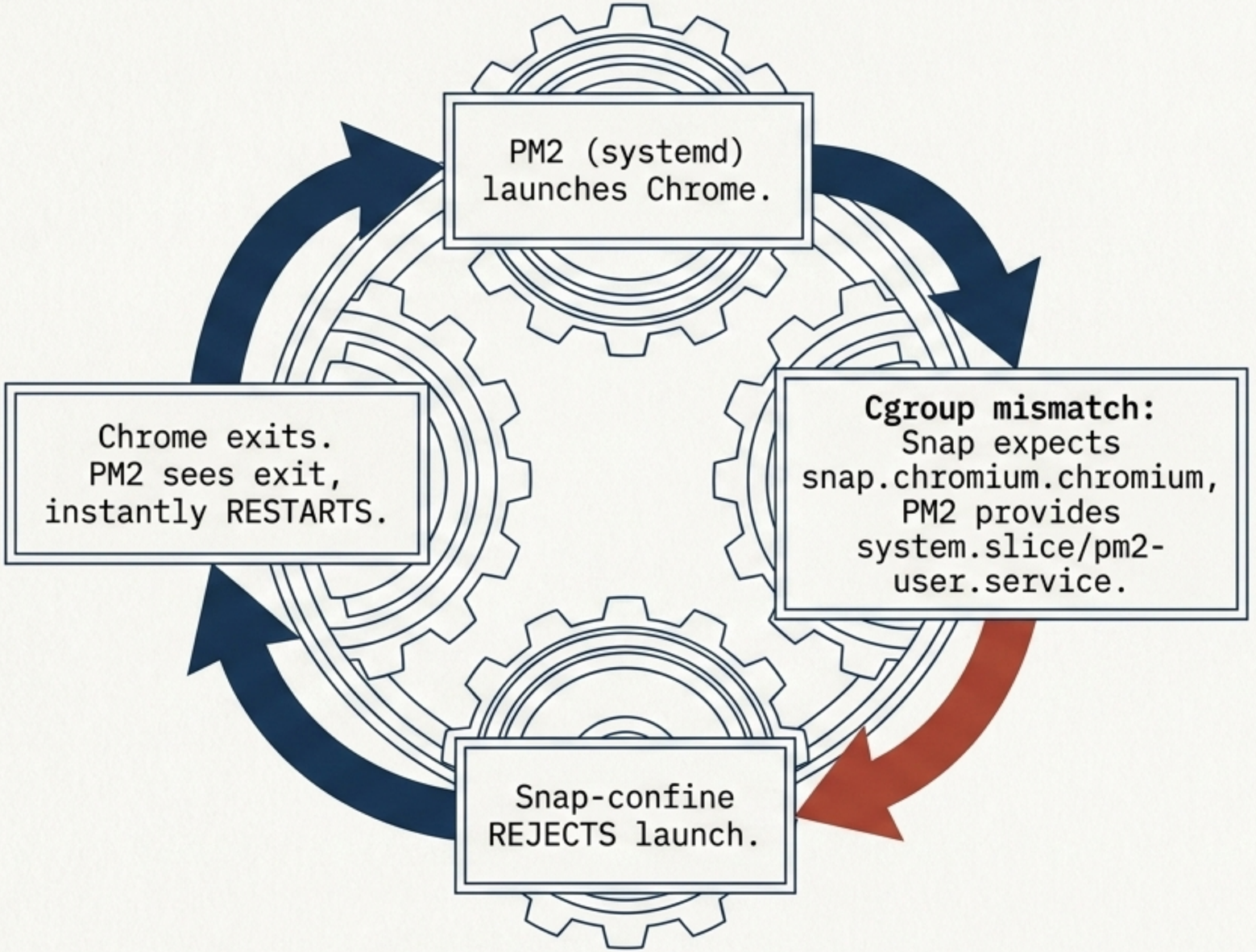
```
Diagnosis: Resource exhaustion. Kernel alive. Userspace dead. Hard reset required.
```

The Methodical Triage Process



Do not guess. Check the infrastructure, then network, then application, then system logs. Numbers and states are more reliable than hunches.

The Mastermind: An Infinite Crash Loop



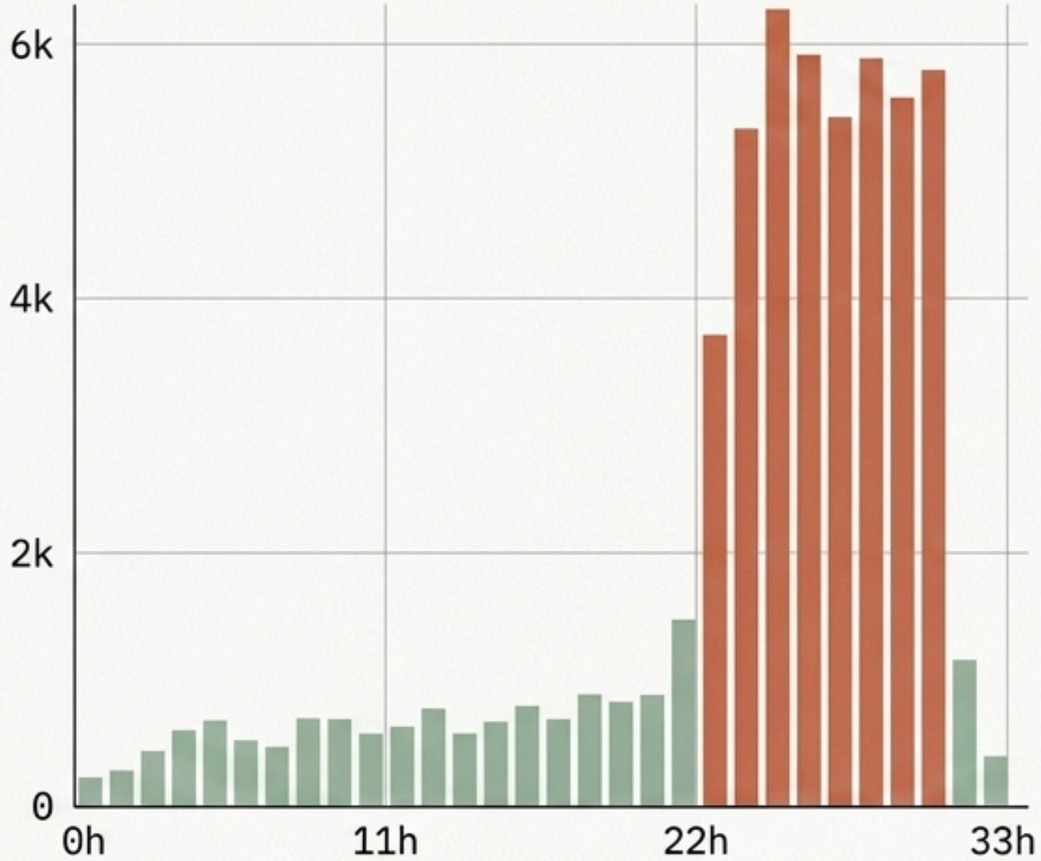
3,055 snap cgroup rejections in 33 hours.

Only 12 launches succeeded (due to race conditions).

Result:
PIDs exhausted, memory exhausted, disk I/O saturated.

The Accelerants

6,066 Failed SSH Login Attempts



Internet-exposed IP. Roughly 3 brute-force attempts per minute over 33 hours. Each attempt spawned an sshd child process, accelerating the fork bomb.

RAM: 100%

Swap Space: 0 GB

16GB RAM with zero swap means zero buffer. When memory fills, the OOM (Out of Memory) killer deadlocks instantly instead of reclaiming space.

The Early Warning Sign

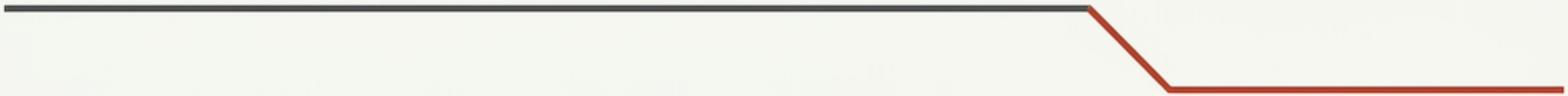
```
rsyslogd: action 'action-1-builtin:omfile' suspended
rsyslogd: action 'action-1-builtin:omfile' resumed
```

When rsyslogd cannot write to its own log files, the system is moments from total freeze. If your monitoring detects the omfile suspended pattern, intervene immediately.

Act 1 Complete: Stopping the Bleeding

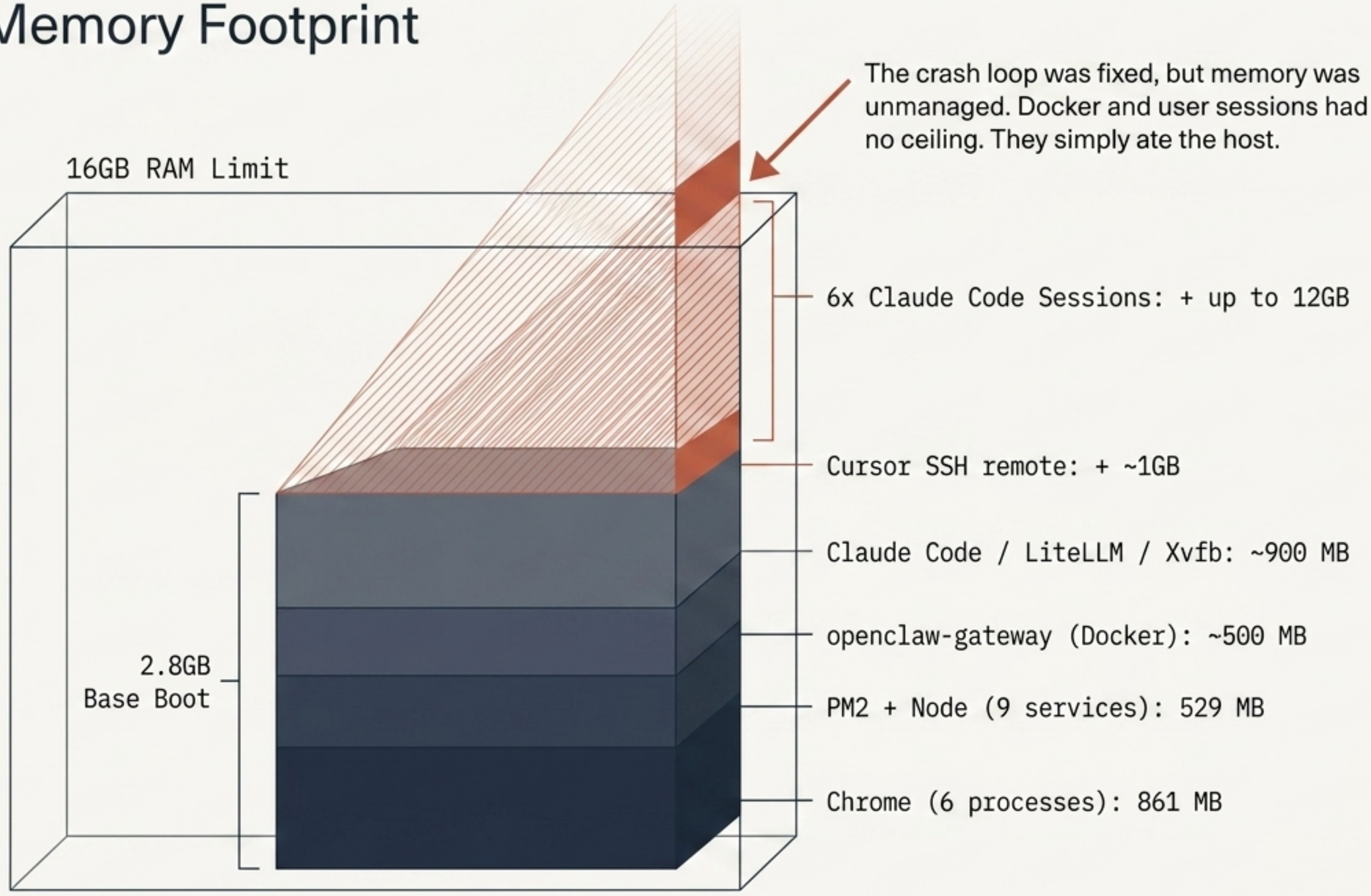
✓	**Replace Snap Chrome.** Installed .deb package. Eliminates the cgroup mismatch entirely.
✓	**PM2 Restart Limits.** Configured max_restarts and restart_delay. Prevents future fork bombs.
✓	**Fail2ban Active.** Banned first malicious IP within 10 minutes.
✓	**4GB Swap Added.** Creates a 25% buffer window for the OOM killer to operate safely.
✓	**SSH Hardened.** MaxStartups 5:50:10. Drops brute force connections at the TCP level before cryptographic load.

Five fixes applied.
The system was stable.

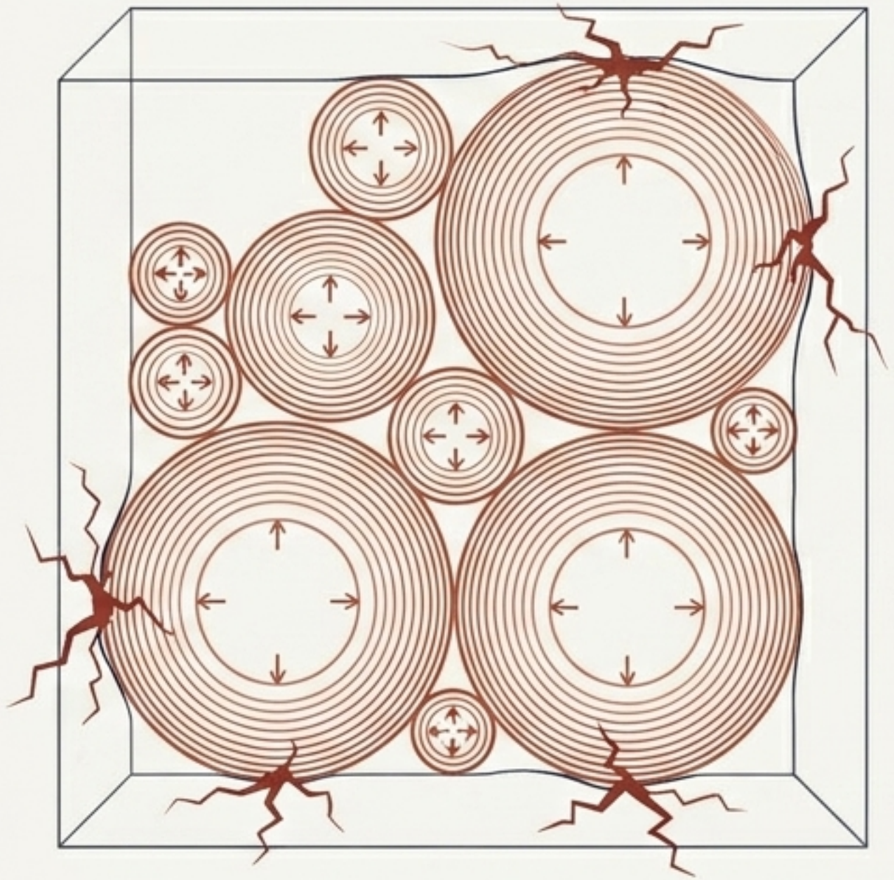


A few hours later…
it died again.

Mapping the Memory Footprint

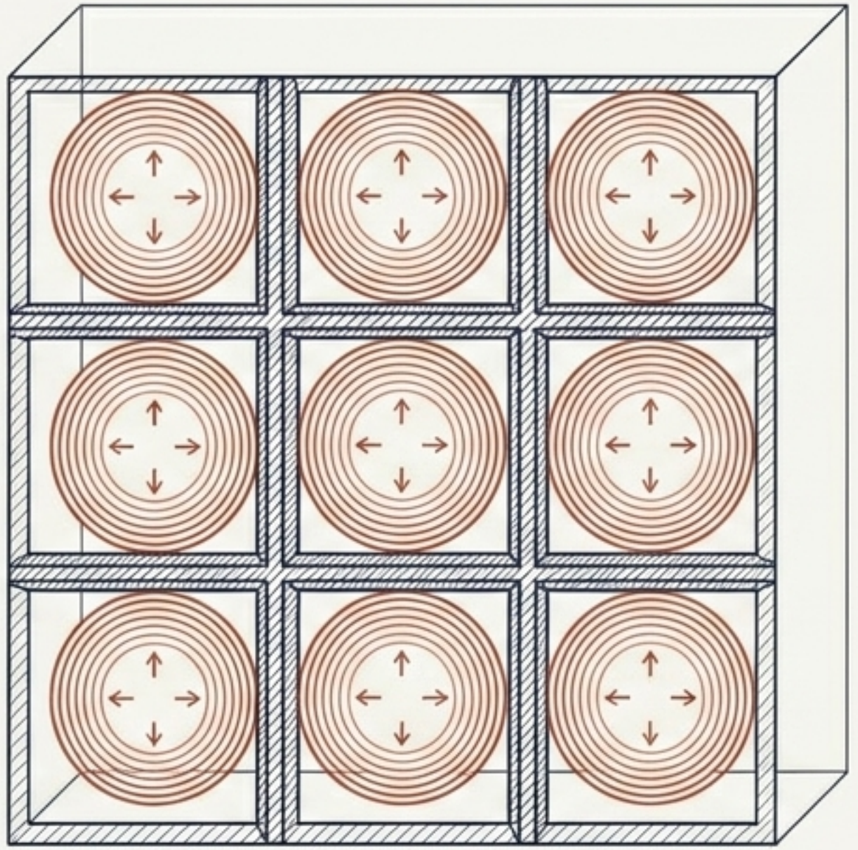


The Real Issue: Unbounded Growth



Fragile - Act 1

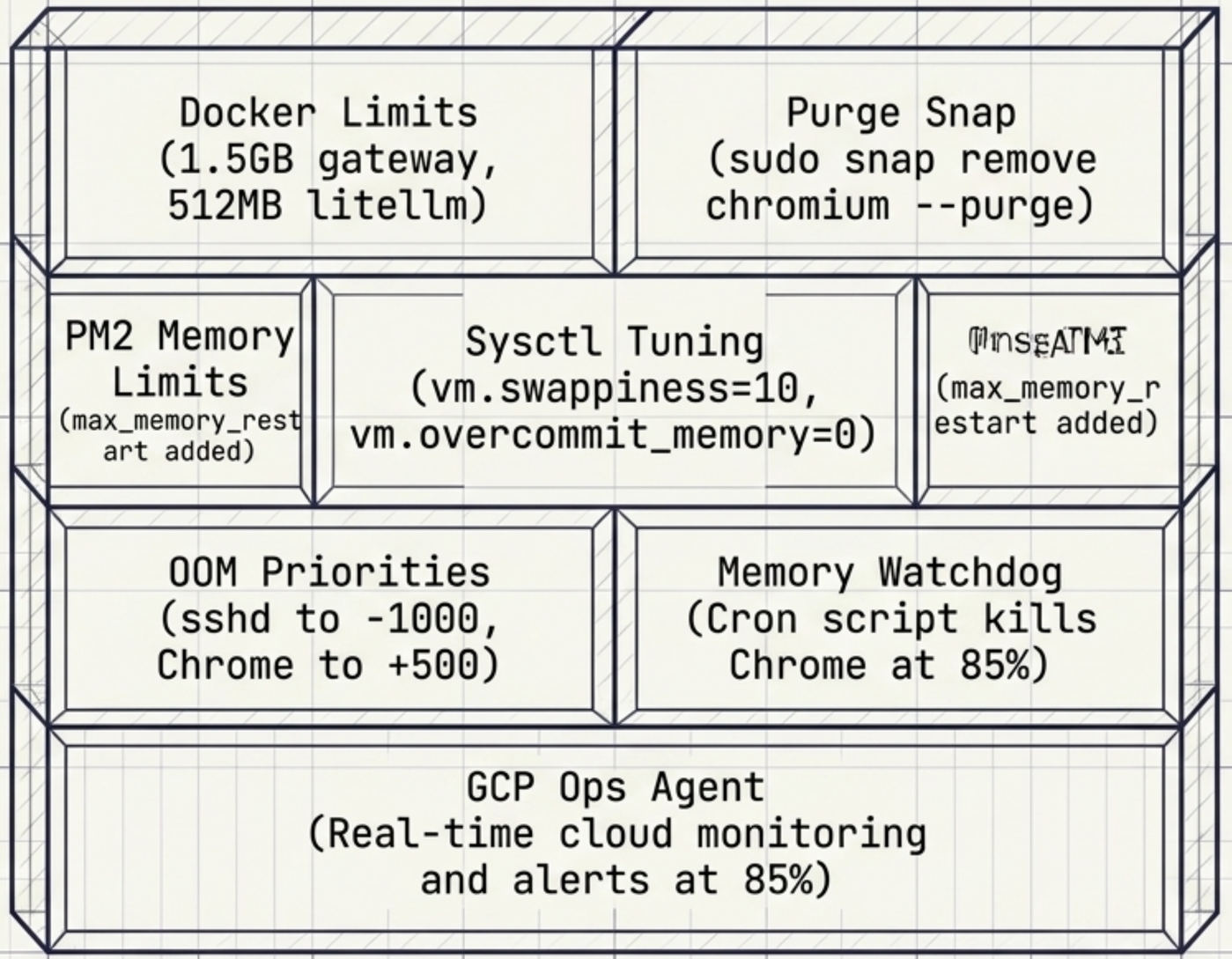
Unbounded. Any single process can consume 100% of host resources, **killing all neighbors.**



Resilient - Act 2

Bounded. Every process has a strict memory ceiling. If a process spikes, only that process **dies and restarts**.

Round 2: Setting Strict Ceilings

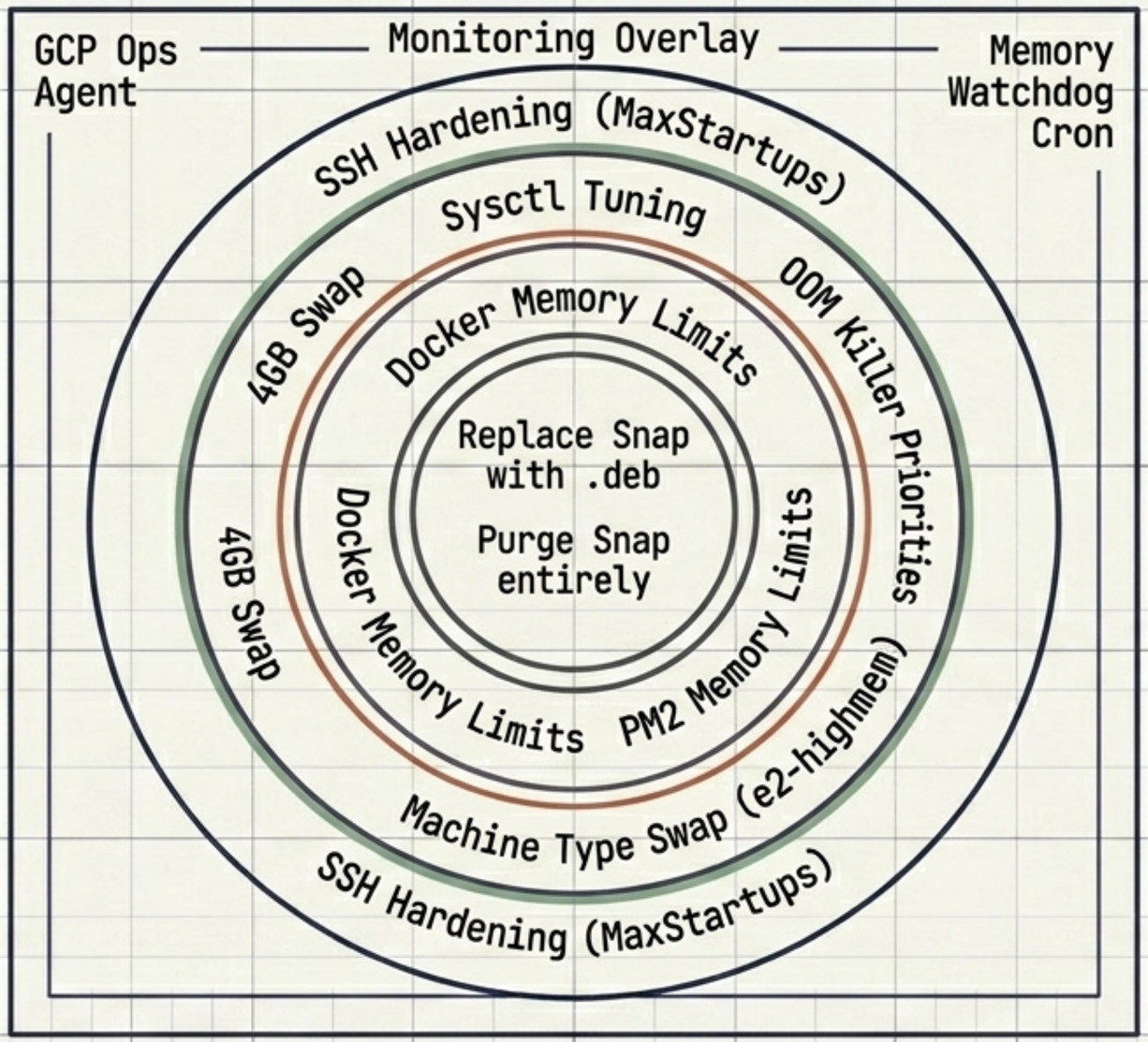


A Case of Mistaken Identity: The Machine Type

Machine Type	vCPU	RAM	Monthly Cost	Workload Fit
c3-standard-4	4	16GB	~$152	Compute-heavy (HPC/Gaming). Wasted money for idle AI APIs.
e2-highmem-4	**4**	**32GB**	**~$131**	**Bursty/General purpose. Double the RAM, lower the bill.**

A 16GB machine cannot handle **6 concurrent 2GB user sessions**. Switching to e2-highmem-4 provided 32GB RAM for $21 less per month, perfectly matching the bursty LLM workload.

The 12-Layer Defense in Depth



The Final Verdict

One bad process should never take down an entire machine.

Ensure strict cgroup confinement, restart limits, and swap space. Treat every process as a **potential threat** to its neighbors.

Fixing the surface is not fixing the root cause.

Eliminating a crash loop is just treating a symptom. Until every process has a defined memory ceiling and real-time monitoring, the system remains **fatally vulnerable**.