Night Watch

BUILDING A WATCHDOG THAT LIVES OUTSIDE THE BLAST RADIUS

One evening, one Raspberry Pi 4, zero monitors attached. This is the engineering companion to Build Note N-010 at michaelgaff.com/journal/night-watch. It documents an incident, the hardening that followed, a fully headless build, and the monitoring architecture that came out of it. Every credential, address, hardware identifier and personal detail has been removed. What is left is the architecture and the technique.

1. The incident chain

A Mac in Indianapolis went dark and stayed dark for five weeks. The operator was in Florida. The outage was a chain, not a single fault, and every link was individually boring.

- **Power failure.** The building lost power. Ordinary.
- **Automatic restart was off.** macOS ships with restart after power loss disabled. When the power returned, the machine simply stayed off.
- **The network failed over to a dead router.** The condo internet moved to its backup path, and the backup device was not functioning. It could not be power cycled remotely, because the only route to it ran through the thing that was broken.
- **The machine was a backup target.** It received a nightly mirror from the production Mac in Florida. For five weeks, that job wrote into nothing.

The real defect was not any of those. It was that nobody outside the room could tell *which layer* had failed. Host, router, or provider: from a thousand miles away all three present identically, as silence. That observability gap is the bug this build fixes.

2. Hardening the host (about ten minutes on site)

First, remove the failure mode entirely. On macOS this is two commands.

```
sudo pmset -a sleep 0 disksleep 0 autorestart 1 womp 1
sudo systemsetup -setremotelogin on
```

- **autorestart 1** is the headline. The machine now powers itself back on after any power failure, which is the exact fault that caused the outage.
- **sleep 0** and **disksleep 0** remove the sleep case, so the host is never merely unreachable because it dozed off.
- **womp 1** enables wake on LAN, a belt and suspenders layer for the same problem.
- **Remote login on** restores the administrative path once the host is up.

Kill the silent variant too

There is a quieter version of the same outage: power is fine, host is fine, and the mesh VPN key expires, so the node drops off the private network and every alias pointing at it goes stale. Disable key expiry on every always on

node in the admin console. Prefer stable DNS names over addresses in scripts, because a node that reauthorizes can rejoin as a new node with a new address and quietly break unattended jobs.

Verification, not vibes

Do not assume the fix. Connect from the traveling laptop through the fixed alias, then run a ping across the mesh from the production machine to the repaired host and confirm zero percent loss. Then confirm the nightly backup path completes end to end before leaving the building.

3. Choosing a watchdog outside the blast radius

The governing rule: **a monitoring system running on the machine it monitors is a diary, not an alarm.** It reports what happened, later, and only if it survived the event.

So the watcher was placed on a different machine, in a different city, on a different power grid, behind a different internet provider from the production host. Correlated failure is the enemy. Physical separation is the cheapest way to break the correlation.

- **Hardware.** Raspberry Pi 4, roughly sixty dollars, already owned. One 128GB microSD delivered same day.
- **Operating system.** Raspberry Pi OS Lite, 64 bit. No desktop.
- **Stack.** Mesh VPN client, Docker, Uptime Kuma, a wake on LAN utility, a Telegram bot for alerting.
- **Total build time.** Roughly two hours including every detour documented below.

4. The headless flash

No screen, no keyboard, no video cable was used at any point. The imaging tool writes configuration directly into the boot partition before the card ever enters the Pi.

- In Raspberry Pi Imager, choose the Lite 64 bit image, then open OS customisation.
- Set the hostname, create the user account, and **enable SSH**. This is the step to double check, because it is the one that is easy to miss.
- On first boot the Pi applies that configuration and announces itself over mDNS, so it is reachable by name on the local network with nothing attached to it.

5. Recovery when the remote access toggle is missed

Symptom. The host resolves and answers a ping in single digit milliseconds, but the connection on port 22 is refused.

Diagnosis. Host is up, service is absent. This is not a network problem and it is not a reason to reflash.

Fix. Power down, move the card back to a workstation, place an empty file named ssh in the boot partition, eject, reboot. The Pi enables the service on boot when it finds that file. Recovery takes about sixty seconds.

```
touch /Volumes/bootfs/ssh && diskutil eject /Volumes/bootfs
```

> Worth internalising: **connection refused is good news during a headless boot.** Refused means the host resolved, answered, and declined. Something is alive. A timeout is the bad outcome. Refused means fix the service. Timeout means go look at the network.

6. Mesh membership and remote hands

Join the watchdog to the mesh VPN so it is reachable from anywhere, then disable key expiry on the node so it cannot silently drop off later. Install a wake on LAN utility and wrap it in a one line script targeting the monitored host, so any device on the mesh can wake that machine through the Pi.

```
curl -fsSL https://tailscale.com/install.sh | sh && sudo tailscale up
sudo apt -y install wakeonlan
echo 'wakeonlan <TARGET_MAC_REDACTED>' > ~/wake-imac.sh && chmod +x ~/wake-imac.sh
```

> One limitation to understand before you rely on it: wake on LAN wakes a sleeping machine on the local segment. It cannot help a machine that is powered but off the mesh. Those are different failures and they need different responses, which is exactly why the monitoring board below separates them.

7. The watcher: one container, nine monitors

Uptime Kuma, self hosted and open source, in a single container. The restart policy survives reboots and the named volume survives container upgrades, which together mean the watchdog needs no attention after the evening it was built.

```
curl -fsSL https://get.docker.com | sudo sh && sudo usermod -aG docker <user>
docker run -d --restart=always -p 3001:3001 \
  -v uptime-kuma:/app/data --name uptime-kuma louislam/uptime-kuma:1
```

The board

- **Seven HTTPS monitors.** The production command center plus six public web properties. Sixty second heartbeat.
- **Two ICMP monitors across the mesh.** One to the production machine, one to the recovered backup host. These are addressed by their private mesh identity, not their public one.

Why the two pings matter more than the seven checks

The mesh pings are a triage layer. On their own the HTTPS checks tell you a site is down. Paired with a ping they tell you *where* it is down, which is the difference between an alert and an answer.

- **Site down, mesh up.** The host is alive and serving. The tunnel, proxy, or CDN in front of it broke. Fix the edge.
- **Site down, mesh down.** The machine itself is gone. Power, network, or hardware. Wake it or go to it.
- **Site up, mesh down.** Cached or edge served content is masking a dead origin. The most deceptive of the three, and invisible without the ping.

That is a differential diagnosis performed from a phone in about four seconds. Before the build it could not be performed at all.

Certificate expiry, free

Every HTTPS monitor tracks certificate expiration with no extra configuration. Within minutes of going live the board flagged one property renewing in forty one days. A failed automatic renewal now pages the operator before a single visitor sees a browser warning. This is the highest value monitor nobody remembers to set up.

8. Alerting: a bot that only speaks when something is wrong

Alerts route to a dedicated Telegram bot, created through BotFather and wired into the monitoring notifier as a default applied across every monitor. The design goal is a channel that stays silent. The first message it sent was a test. Ideally it is also the last message for a very long time.

Three gotchas that cost real time

- **HTTP 401 on the test send.** The token is wrong, and it is nearly always a copy that lost a character. Recopy the full token from the bot listing, do not retype it. Rotating the token invalidates the old one immediately.
- **Chat not found.** A bot cannot initiate contact, and the update feed only shows chats that have already messaged it. Send a start command to the bot first, then let the notifier auto fill the chat identifier. Verify you are messaging the exact bot you created, because searching by display name can land you on a stranger's bot.
- **Hygiene.** Tokens are passwords. Any token that appeared on a screen, in a screenshot, or in a chat log is burned. Revoke it, paste the replacement, resave the notifier.

> Also confirm the two settings that quietly decide whether alerting works at all: the notifier must be marked as default enabled, and it must be applied to all existing monitors. A notifier that is configured but not attached is the most common reason a board looks healthy and never pages.

9. Resulting architecture

- **Florida.** Production machine runs the command center, the bots, and the scheduled jobs. It mirrors itself nightly over the mesh to the backup host in Indiana.
- **Indiana.** The backup host now restarts itself after power loss and accepts remote login. The Pi watches all seven public properties plus both machines, from a separate power grid and a separate provider, and can wake the backup host on demand.
- **Anywhere.** The traveling laptop and a cellular tablet reach every node over the mesh, so a dead condo connection no longer means blindness.
- **Alerting.** Any failure pages a phone within roughly sixty seconds.

Single points of failure removed in one evening: the unmonitored power state of the backup host, the unmonitored production machine, and the unanswerable question of whether the remote site had internet at all.

10. Field notes

- **Paste blocks lose races against password prompts.** If line one triggers a prompt, the remaining lines are consumed as password guesses. Symptom: repeated authentication failures with your own commands as the passwords. Rule: one command at a time near any prompt.
- **Apostrophes in shell comments can wedge the shell.** A pasted comment containing a contraction opens an unterminated quote and the terminal waits forever. Rule: paste blocks ship without comments.

- **Placeholders in handed off commands will be pasted literally.** Angle bracket placeholders end up in the command exactly as written. Rule: hand off only fully resolved commands.
- **Local discovery names and mesh names are different layers.** The mDNS name works on the local network. The mesh name works from anywhere. Know which one you are using, because the one that works at the kitchen table is not the one that works from a hotel.
- **A USB drive that passes no power is a cable or enclosure suspect first.** Same day symptom onset, no activity light, no passthrough. Replace the cheap part before condemning the disk.

11. Failure modes and the layer principle

Six days after the build, the same backup host dropped off the network again. Different cause: the mesh client updated itself and required reauthorization, and on reauthorization the host rejoined as a new node with a new address. Every script and alias pointing at the old address was addressing a machine that no longer existed.

- **What caught it.** The backup job reported red, and the monitoring board identified which layer had failed and, just as usefully, which layers had not.
- **The repair.** Repoint the alias to the current node, accept the new host key interactively (an unattended job cannot answer a host key prompt, so it will fail silently until a human accepts it once), restart the job by hand, and confirm a clean full snapshot.
- **Why nothing was at risk.** A secondary nightly job copies the critical files to cloud storage on an independent path. It ran exactly as designed while the primary path was broken.

That is the layer principle, and it is the whole thesis. A night watch is not one component that never fails. It is several independent components that fail at different times, for different reasons, on different infrastructure.

Hardening still queued

- Accept new host keys automatically on the unattended path so an address change cannot silently break a nightly job.
- Network wide DNS filtering on the Pi, with an override on the mesh and a fast rollback.
- A subnet router so the remote LAN, including router administration, is reachable without a trip.
- Environmental sensors and a smart plug that power cycles the router when it stops answering, handled locally with no internet required. A router that heals itself.
- An uninterruptible power supply on the Pi and the router, so the exact blip that started this story cannot repeat.

12. The thesis

Observability is not a big company luxury. A one person operation with production systems in one state and a life in two got monitoring in a separate state, automatic recovery from power loss, remote hands, and a bot that texts, in one evening, for the price of a nice dinner, on hardware that was already sitting in a drawer.

The unlock was not the hardware. It was that the entire build ran headless and conversational: flash a card, paste a command, read the error out loud, fix it, keep moving. The machine never needed a screen, because the operator already had one.

Companion to Build Note N-010, published 25 July 2026 at michaelgaff.com/journal/night-watch. Written for engineers. All credentials, hardware identifiers, private addresses and personal data have been removed.