**Engineer** Platform

# Reliability & Backups

43 achievements

## Connect the parts of your system

The work collected under “Reliability & Backups” shows one part of a working system. We can help connect that part to the rest of your operation. Monitoring and alerts that speak up while a fault is still small, and the figures that show how often it stays that way.

**Explore our services** Get in touch →

## Find a comparable challenge

Follow an example that resembles your situation. Its review describes the decisions behind the result, beyond the technology label. Built the company's first observability dashboard — the company's first view of its own systems

**Built the company's first observability dashboard** Get in touch →

### Built the company's first observability dashboard

Developed and launched the company's first observability dashboard, providing real-time system performance insights and data visualization on the large office TV.

### Ran 10 PostgreSQL and MS SQL servers on Ubuntu

Designed, deployed, and maintained 10 PostgreSQL and MS SQL servers on Ubuntu Linux VPS, ensuring optimal server performance and reliability.

### Automated GIS SaaS deployment with GitHub Actions

Automated GIS SaaS application deployment, data processing, and reporting system using GitHub Actions CI/CD, Python, Bash, and SQL.

### Automated delivery of 20 GIS data pipelines

Automated delivery of 20 GIS data pipelines and app data ETL processes, streamlining infrastructure automation and reporting.

### Automated 100 critical data backups using Barman

Automated 100 critical data backups using Barman, Google Cloud, Bash, and Python, ensuring data integrity across databases.

### Ran 20 Docker apps on Google Cloud and AWS

Deployed and maintained 20 Docker containerized applications, troubleshooting with Podman and Kubernetes, and managing R-based apps on Google Cloud and AWS.

### Managed 30 Ubuntu Linux VPS instances

Managed 30 Ubuntu Linux VPS instances, implementing disaster recovery strategies and ensuring optimal network configurations.

### Cut operational risk with Grafana and Prometheus

Mitigated operational risks by implementing a monitoring dashboard using Grafana and Prometheus, improving system reliability.

### Managed 8 WireGuard and IPsec VPN connections

Managed and troubleshot 8 WireGuard VPN and IPsec VPN connections, ensuring secure communication across Google Cloud and Linux systems.

### Administered network infrastructure

Administered network infrastructure for over 1,000 servers, ensuring optimal system deployment, security, and troubleshooting.

### Automated SSL/TLS certs for 100 Docker apps

Automated SSL/TLS certificate creation for 100 Docker applications, ensuring secure connections across Ubuntu Linux hosts.

### Streamlined CI/CD processes, saving 4,000 hours

Streamlined CI/CD processes, saving 4,000 hours by introducing automation in software development pipelines.

### Configured and deployed 1,000 Wi-Fi routers

Configured and deployed 1,000 Wi-Fi routers, improving network accessibility and performance for clients.

### Administered 100 Bare Bone servers

Administered 100 Bare Bone servers, physical networks, and IP telephony systems, ensuring robust infrastructure for company growth.

### Administered 40 websites on Apache and Nginx

Administered 40 websites on Ubuntu Linux hosting servers with Apache and Nginx, ensuring high availability and performance.

### A mapping product, carried end to end

Architected, developed and supported infrastructure, data processing and a map application over 2 years, covering both software delivery and operational maintenance.

### Generated the API contract from the database out

Generated the API contract outward from the database — OpenAPI, a 44,076-line typed TypeScript client, 61 mock handlers and the limits the UI enforces — with a guard at every hop that fails on drift.

### Built email with failover and delivery logging

Built email as a platform capability — three providers with failover, delivery webhooks, send and delivery logging, templating and campaigns — behind a startup check that will not boot without one.

### Provisioned Azure infrastructure as code

Provisioned Azure infrastructure as code with Bicep — Container Apps, PostgreSQL Flexible Server, Front Door/WAF and networking — across the development, staging and production environments.

### Built GitHub Actions CI/CD pipelines

Built GitHub Actions CI/CD pipelines with a distroless production frontend image and multi-environment promotion.

### Owned end-to-end platform deployments to Azure

Owned end-to-end deployments of the platform to Azure, managing releases across development, staging and production environments.

### Implemented backups and disaster recovery as code

Configured database backup retention as infrastructure-as-code, then audited the recovery position and documented the restore procedure — naming the remaining gaps rather than leaving them to be found during an incident.

### Ran 24/7 support for ~1,000 IPTV/OTT servers

Provided round-the-clock 24/7 infrastructure support for an IPTV/OTT streaming platform, administering ~1,000 servers plus client-owned systems for global customers in China, the US and Germany.

### Kept IPTV streaming signals running non-stop

Ensured uninterrupted delivery of IPTV streaming signals between suppliers and clients, monitoring and maintaining the streaming network and IP telephony around the clock.

### Built infrastructure that still runs years later

Planned and implemented new infrastructure functionality for internal and external systems, building solutions durable enough to still run years later with minimal change.

### Built a test suite across Go, SQL and browser

Built a layered automated test suite — 981 Go tests, 543 frontend and browser specs, 494 SQL behavioural tests — with mutation testing, property-based tests and an accessibility gate.

### Moved slow work onto a River job queue

Moved slow work off the request path onto a River job queue — 15 worker modules, 8 scheduled tasks and 20 pg_cron jobs — so a request returns while the work behind it carries on.

### Built first-party error monitoring and tracing

Built first-party error monitoring and OpenTelemetry tracing rather than buying them — payload sanitising, spike and regression detection, symbolication and a synthetic heartbeat — behind 11 operator views.

### 1,022 migrations behind a schema drift gate

Kept the schema honest across 1,022 migrations with a CI gate that builds the database both ways — a fresh install, and an install plus every migration — and fails when the two disagree.

### Built fail-closed rate limiting and anti-abuse

Built fail-closed abuse controls — 22 Redis-backed rate limiters, Cloudflare Turnstile, request idempotency and an origin lock — so the platform sheds bots and floods instead of trusting its callers.

### Ran a continuous security scanning programme

Established a continuous security programme — code scanning, DAST, dependency and vulnerability checks, SBOM generation, secret scanning and SHA-pinned actions — alongside 21 written security audits.

### Infrastructure defined in a repository

Built the company's own infrastructure as 19 Ansible playbooks and 34 roles across 12,065 lines of YAML, converging a live host to a declared state with every play idempotent.

### Public services on a small server

Hosted the company’s public services on a single-core 512 MB host — a git forge, a web server for seven domains, Tor, two alternative-protocol servers and intrusion banning — treating 464 MB of usable memory as the design constraint; backup automation is a separate implementation.

### Proved the intrusion ban works every run

Proved the intrusion-banning path end to end on every hardening run by banning a reserved test address, reading the resulting kernel rule and unbanning in a guaranteed cleanup block, so a jail that stops working fails a run instead of reporting healthy.

### Backup automation with an explicit restore method

Implemented restic backup automation with retention, integrity checks and a monthly restore-test workflow; documented storage activation and live-restore verification as separate operational steps.

### Built dead-man's-switch host monitoring

Built dead-man's-switch monitoring that pings only while memory and disk are healthy, so a degraded host raises an alert by going silent — and caught six variable names saying "free" where the check correctly measured "available", an order of magnitude apart on a 464 MB box.

### Made Ansible check mode tell the truth

Made check mode tell the truth across the whole platform after finding six probes deciding on a value the host never gave, because Ansible's command module reports success under --check while skipping the command entirely.

### Added a one-second preflight for secrets

Added a preflight play that runs the same code as the converge against operator-local secrets in about a second, after a half-applied production run died on its ninth task with the swap settings already written to the live host.

### Split every operational secret per host

Split the four per-host operational secrets after establishing that two hosts sharing one dead-man's switch alert less than two switches, not more, and that a shared backup passphrase can expose both hosts' backups, even in separate repositories, if someone obtains it and can read both repositories' encrypted data and key files.

### Held the generator to 981 tests and 92% coverage

Held the generator to 981 test cases at a 92% branch-coverage floor with warnings treated as failures, and asserted idempotence by running the whole build pipeline twice from an empty file and requiring the second pass to change nothing.

### A bounded event pipeline for a macOS app

Bounded event buffers and logging after a recorded macOS incident with 41 MB per second growth and 111 GB of compressed pages on a 36 GB machine; a recorded full test run reduced core log lines from 610,996 to 1,411.

### Reached the unused half of the core

Reached the half of the messaging core the application had never used — backup transfer, disappearing messages, message editing and resending, verified invitations, proxies and encryption policy — driving every test against the real library with no mocks.

### Made six collectors state what they skipped

Made six reporting collectors state the part of their input they never read — files that stayed shut, probes not installed, journal lines unrecognised — and fitted thresholds to 116 measured samples, after establishing that a blind collection and a healthy host render the same page.

### Focus the engineering effort

Bring the deployment path, the operational risks and the people on call. We can scope a change with checks, a recovery route and clear ownership.

**See what it costs** Get in touch →

## Put the subject in your context

Tell us how “Reliability & Backups” appears in your own system. We can help choose the relevant service and define the work around it.

**Explore our services**

welcome@engineer.company +45 50 10 61 84 Copenhagen, Denmark

Take this page with you

Contact card

**Carefully Crafted With Love by D.**

**CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N**

**Copyright © 2025 – present · Engineer ApS · All rights reserved.**

**Printed from https://platform.engineer.company/categories/reliability/**