Engineer Platform

Linux & Servers

27 achievements

Connect the parts of your system

The work collected under “Linux & Servers” 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.

Wrote 1,000 RBAC rules to secure Linux servers

Enhanced data security by implementing 1,000 RBAC rules for developers, application instances, PostgreSQL, MS SQL, and other Linux servers, preventing unauthorized access; documented with Ansible automation.

Managed 30 Ubuntu Linux VPS instances

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

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.

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.

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.

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.

Three locks on the door, none of them locked

Found and closed three SSH brute-force protections that had never worked: a ban jail watching port 22 while the daemon listened on 1986, a rate limit shadowed by a broader rule above it, and a ban action whose binary never resolved, so no ban had ever applied.

Hardened SSH to 24 validated directives

Hardened SSH to 24 asserted directives with three-stage validation — the candidate file, the assembled config, then the daemon's own read-back — after the read-back caught the running server silently overriding two of the twenty-four.

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.

Verified firewall rules by position

Verified firewall rules by position rather than presence, reading the numbered rule list and the live packet-filter chain, because a rule that exists is not a rule any packet reaches.

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.

Cut systemd sandbox exposure on every unit

Cut systemd sandbox exposure across every unit this platform installs — a dead-man's-switch service from 9.6 UNSAFE to 1.5, an internet-facing git forge from 8.3 EXPOSED to 1.5 — and added a converge-time parser check after finding a misspelled directive silently ignored in three unit templates.

Built 7 read-only host reporting roles

Built seven read-only reporting roles that render a live host to Markdown — facts, access, git, metrics, traffic, security and provider inventory — under a rule that no number is printed the run did not measure.

Your code on your own server

Deployed the company's own git forge on Soft Serve, private by default with no web panel and its SSH port bound to loopback behind a jump host, and made the landing page in front of it a build artefact of the main site rather than a hand-kept copy.

Ran containers on Podman and Quadlet

Deployed a container plane on Podman and Quadlet under systemd rather than Docker, because Docker's default bridge-port publishing can bypass UFW rules — and gave deploys an unprivileged user with one fixed command instead of root.

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.

Rehearsed the forge CI hook, found 2 defects

Rehearsed the forge-side CI hook and found two defects unreachable by reading the file: a fallback that put an unresolvable argument on the hook's input, and git's own environment variable following the gate into the checkout and turning 19 tests red.

Published a readable Tor onion mirror

Published the site as a Tor onion mirror on a readable address beginning engineer, with the key generated off the host so it never reached this repository or a laptop, and left it the one front door of four that is never counted.

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 “Linux & Servers” 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/linux/