A Raspberry Pi Edge-Computing Cluster for Real-Time Threat Detection

On-sensor object detection • Distributed SeaweedFS storage • MPI & HPL scaling on a ten-node cluster

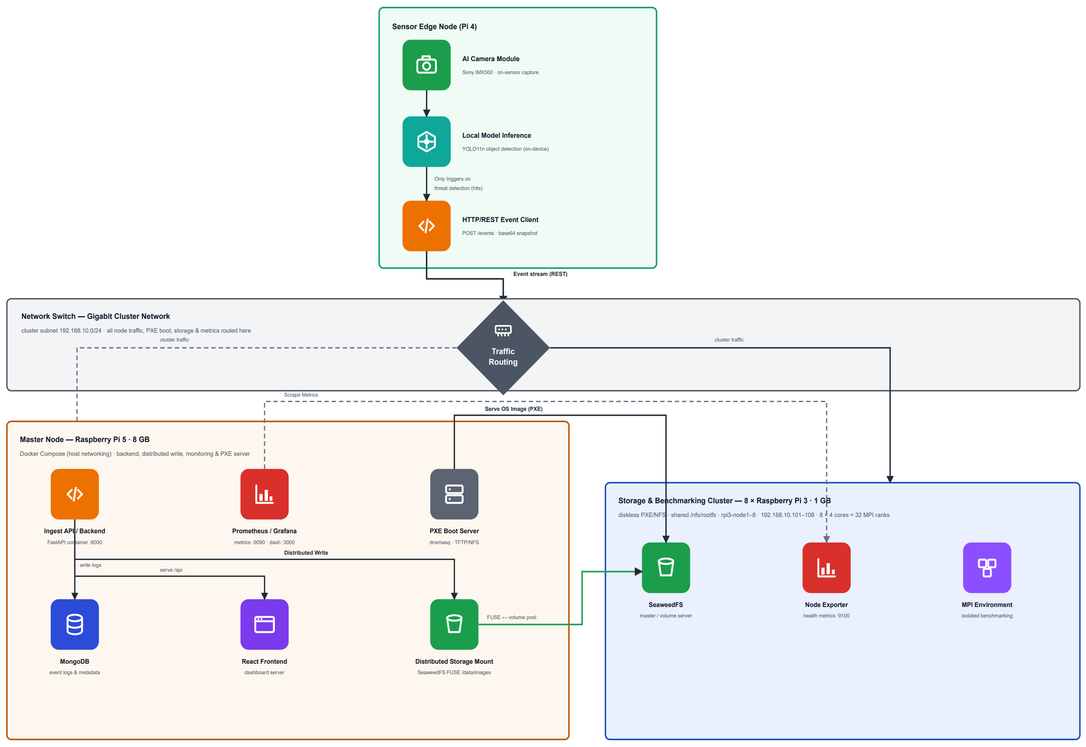
Zeeshan Bashir • Hamza Asaad • Danish Ali • Muhammad Hamza Laghari • Akshit Sandeep Bane • Sahil Qasim Khan • Abdul Rehman • Muhammad Hamza Bilal Baig
Department of Computer Science and Engineering, Frankfurt University of Applied Sciences, Germany

Cloud Computing
SS 2026
Prof. Dr. Christian Baun
Group 3 (SentriPi)

Introduction & System Overview

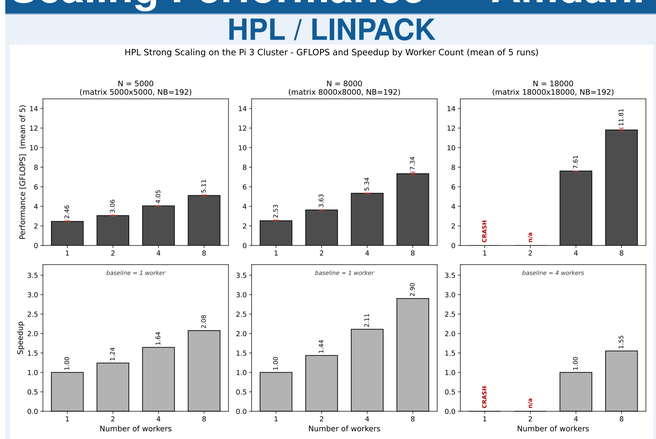
We present a self-contained edge-computing cluster built entirely from commodity Raspberry Pis. A **Pi 5** head node network-boots **eight diskless Pi 3 workers** over PXE/NFS, while a **Pi 4** equipped with a Sony **IMX500 AI Camera** performs object detection *on the sensor* to flag safety-critical events (fire, smoke, weapon, person, mask). Every detection is pushed to a containerised backend, persisted in a **SeaweedFS** object store distributed across the eight workers, and surfaced through a live React dashboard, a Grafana monitoring stack and a Telegram alert bot. The same worker pool doubles as a **high-performance-computing testbed** on which we benchmark HPL, MPI and a distributed render engine to characterise the cluster's strong and weak scaling under Amdahl's and Gustafson's Laws — a low-cost (≈€1100) yet complete demonstration of the promise *and* the limits of Pi-class edge clouds.

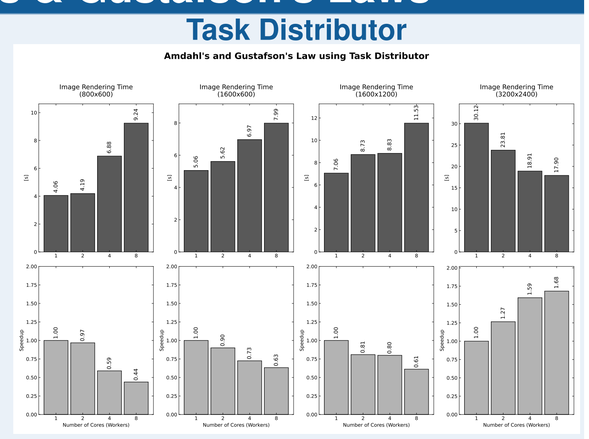
System Architecture — End-to-End Data & Event Flow



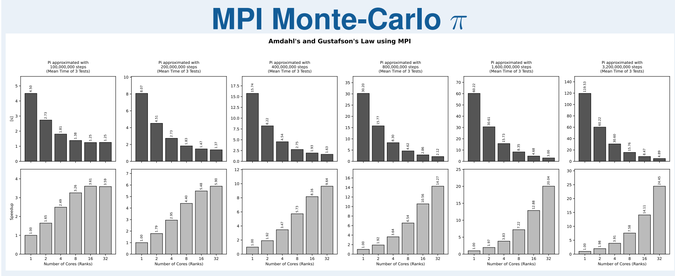
10 nodes on a private Gigabit LAN (`192.168.10.0/24`), containerised with Docker Compose — **1× Pi 5** head (PXE/NFS boot, FastAPI, MongoDB, SeaweedFS filer, Prometheus/Grafana) · **8× Pi 3** diskless workers (SeaweedFS volumes + MPI ranks) · **1× Pi 4** IMX500 camera node.

Scaling Performance — Amdahl's & Gustafson's Laws

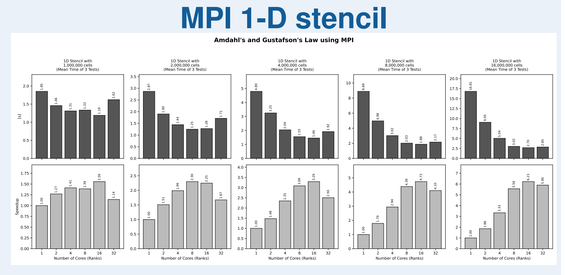




MPI Monte-Carlo π



MPI 1-D stencil



Peak 11.81 GFLOPS on HPL (8 workers, N=18000; larger N scales best). MPI Monte-Carlo π reaches **24.45× at 32 ranks** for 3.2 B steps — textbook **Gustafson** — while small problems saturate (**Amdahl**). The communication-bound stencil **collapses to <0.6× at 32 ranks**, and the Task Distributor render exceeds unit speedup only on the large 3200×2400 image (**1.68×** at 8). **Takeaway:** compute-bound, low-communication jobs scale well; communication-heavy jobs are bounded by Gigabit-Ethernet latency between Pi nodes.

Cluster Provisioning — PXE/NFS Diskless Boot

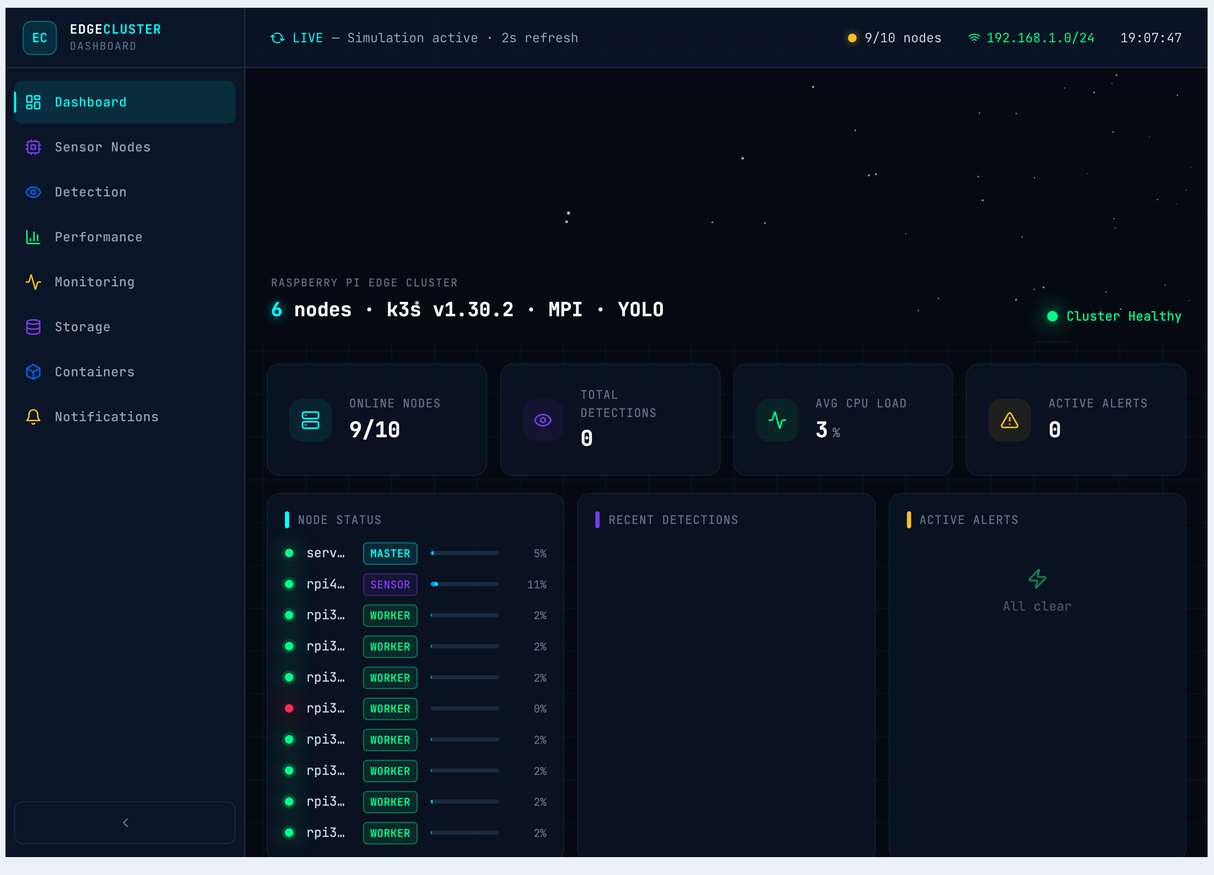
- The **Pi 5 head** runs **dnsmasq** (DHCP + TFTP) and serves the OS image over the LAN — all **eight Pi 3 workers boot diskless** via PXE, no OS on local storage.
- Workers share one **NFS root** (`/nfs/rootfs`): `rpi3-node1-8` at `192.168.10.101-108` — a single image to maintain; a new node joins on power-up.
- Entire service stack orchestrated with **Docker Compose** from the head node.

Distributed Storage — SeaweedFS

- SeaweedFS **master + filer** on the Pi 5; each Pi 3 runs a **volume server** on its local microSD, aggregated into **48×1 GB volume slots**.
- Snapshots written through a FUSE mount (`/data/images`); event metadata indexed in **MongoDB**.
- Objects spread across all eight workers ⇒ horizontal capacity and resilience on commodity storage.

Cluster Monitoring & Live Dashboard

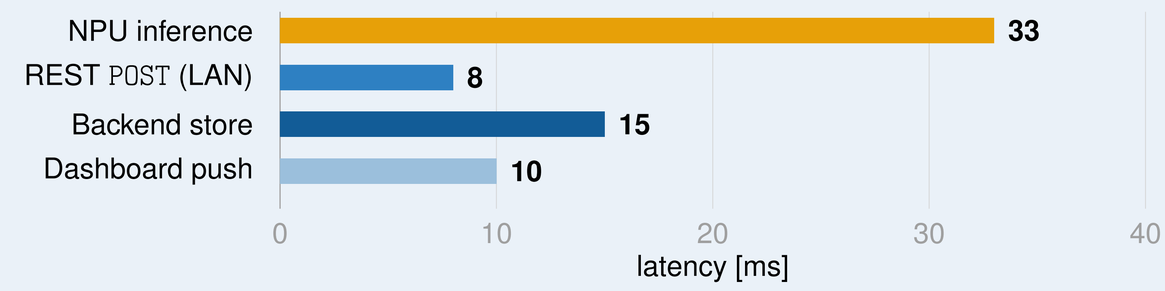
`node_exporter` on all 10 nodes → **Prometheus** (15 s scrape) → **Grafana** dashboard 1860: per-node CPU, RAM, network, disk and temperature. The **React 18 + TypeScript** dashboard renders the live cluster map, detection feed and service health.



Live operations dashboard — 10/10 nodes healthy, per-node status and detection feed.

Backend, API & Alerting — Object → Alert

The **FastAPI** backend exposes `POST/GET /events`, `/metrics/nodes`, `/storage` and `/services`. Each detection is persisted (**MongoDB** metadata + **SeaweedFS** snapshot), streamed to the React dashboard, and pushed to operators through a **Telegram alert bot**.



Object → dashboard ≈ 66 ms on-cluster • → **Telegram alert** + ~0.3–0.5 s (external network)

Edge AI — On-Sensor Object Detection

- Custom **YOLO11n** fine-tuned on Roboflow data plus *hard negatives*; detects **fire, smoke, weapon, person, mask**.
- **INT8**-quantised with Sony's Model Compression Toolkit (`network.rpk`), executed on the **IMX500's on-sensor NPU at 30 fps** — *zero* inference load on the Pi 4 CPU.
- `detector.py` **POSTs each event** (label, confidence, timestamp, snapshot) over REST to the backend.

Software & Tooling

Edge AI: YOLO11n, IMX500 MCT. **Backend:** FastAPI, MongoDB, SeaweedFS, Prometheus, Grafana, Telegram. **Frontend:** React 18, TypeScript, Vite. **Cluster:** Docker, PXE/NFS, OpenMPI, HPL/LINPACK.

NodeManager docs