Cloud Computing

Allgemeine Informatik - Sommersemester 2026

FRANKFURT UNIVERSITY OF APPLIED SCIENCES

Architecture

Hardware and Network Architecture:
The cluster contains a Raspberry Pi 5 head node, eight Raspberry Pi 3 worker nodes, and a TP-Link switch
The switch edge ports use STP PortFast to prevent spanning tree delays during network boot

Centralized Storage and Boot Sequence:
The worker nodes perform network boots via local MicroSD cards that exclusively contain the firmware bootcode.bin
The head node uses DHCP, TFTP, and NFS to provide boot files and root filesystems from an external USB storage device
This architecture centralizes storage and eliminates the need for local operating system installations on the worker nodes

Remote Administration and Security:
The head node routes the internet traffic of the workers via Network Address Translation (NAT)
Tailscale is run on the head node to establish encrypted remote SSH access
This configuration isolates the worker nodes in a private network and prevents public internet sharing

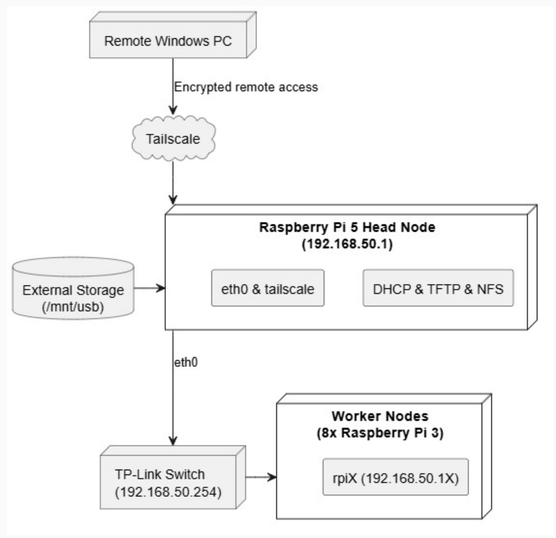

Figure 1:Architecture & Remotemanagement

Frontend & Backend

Backend (Docker Swarm Stack): FastAPI: REST interface for event processing and node management
PostgreSQL: Relational storage of event and node metadata
SeaweedFS: S3-compatible, distributed object storage for image data
Scalable deployment via manager and worker nodes in the Swarm cluster

Frontend (React Dashboard): Central visualization of threat events, camera status, and statistics
Flexible deployment as a Docker container (Nginx) or via Kubernetes (k3s)
API connection to the backend (default port 8000) with a configurable fetch interval

Monitoring

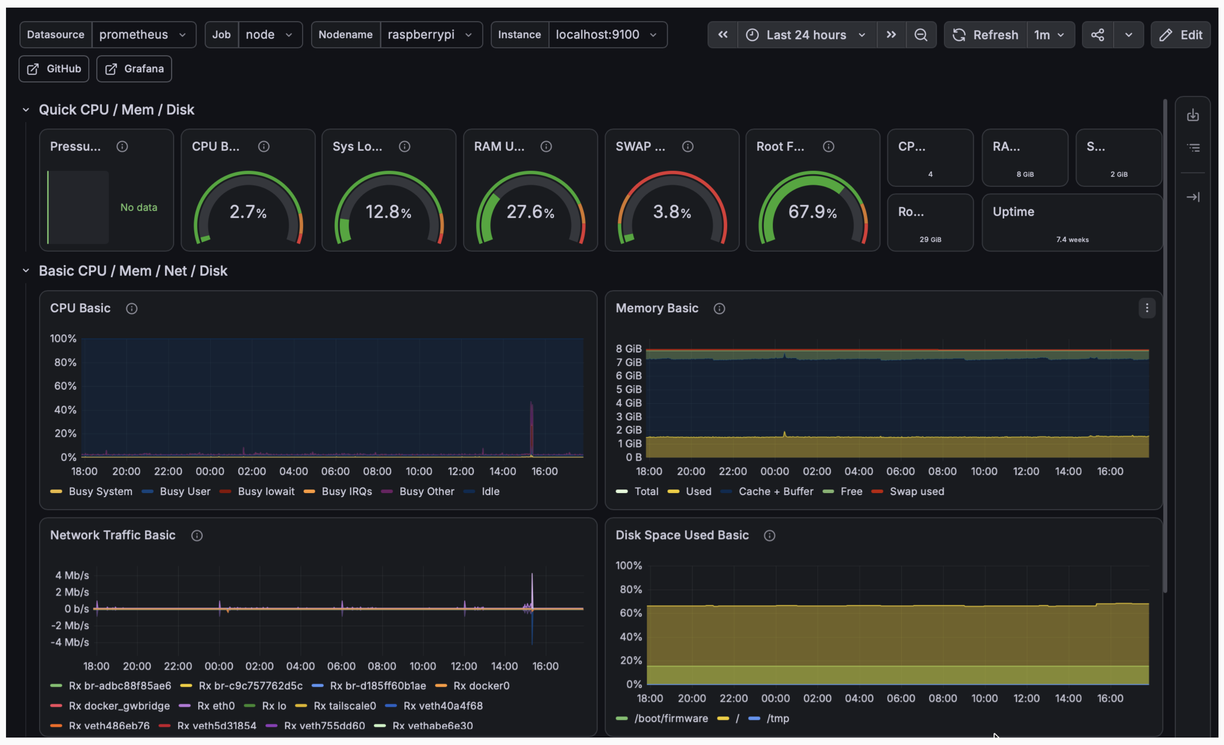

Figure 2:Monitoring

Object Detection & AI Model

AI Model & Architecture: YOLOv11 (Nano) for real-time edge inference. The project compares two platforms: Raspberry Pi 4 (IMX500 AI Camera) and Raspberry Pi 5 (Hailo-8 AI HAT+)
Threat Detection (Classes): Identification of four specific threats: *Fire*, *Mask*, *Scissors*, and *Knife*
Training & Data (Roboflow): Custom dataset merged with public datasets, heavily augmented (rotation, brightness, mosaic), and split 70/15/15
Deployment 1 (IMX500): Model exported to `.imx` via Sony MCT for direct, resource-efficient on-sensor inference
Deployment 2 (Hailo-8): Compilation via ONNX into `.hef` format for high-performance hardware acceleration
Alerting: Real-time event notifications sent to the backend and integrated with a custom Telegram Bot

Authors

Yazan Abou Samra, Lorenz Apel, Hicham Chatir, Algis Cinar, Ibrahim El Haj Moussa, Michael Morgenstern, Mohammed Ali Salem Al-Qafeai, Christopher Scheidler Aguilar, Deniz Zimmer [1]

Benchmark Results

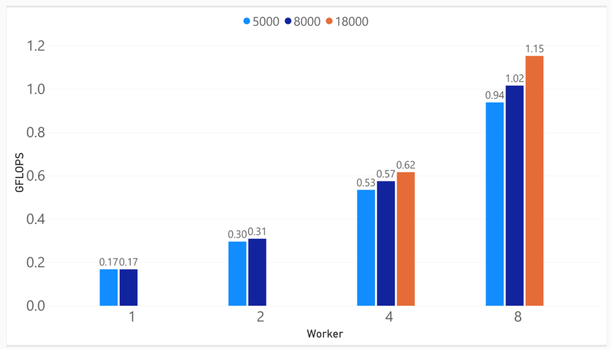

Figure 3:HPL/LINPACK Performance

Figure 6:HPL Strong Scaling

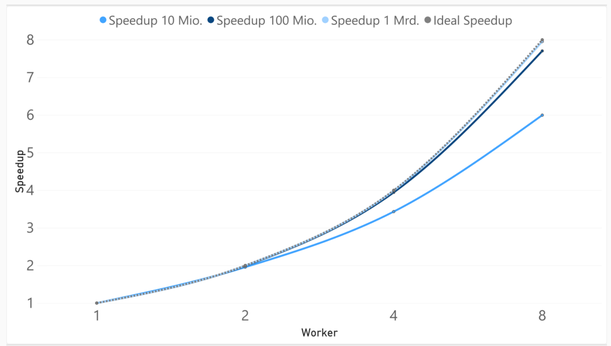

Figure 4:Monte Carlo Strong Scaling

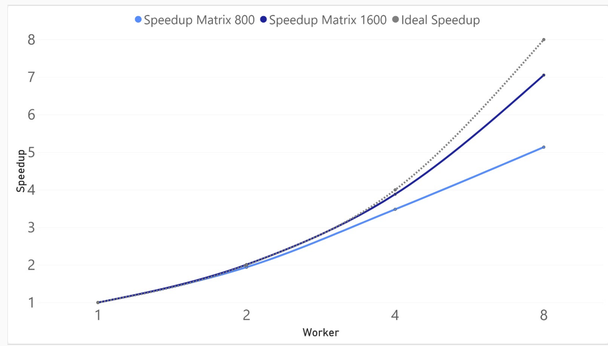

Figure 7:Matrix Multiplication Strong Scaling

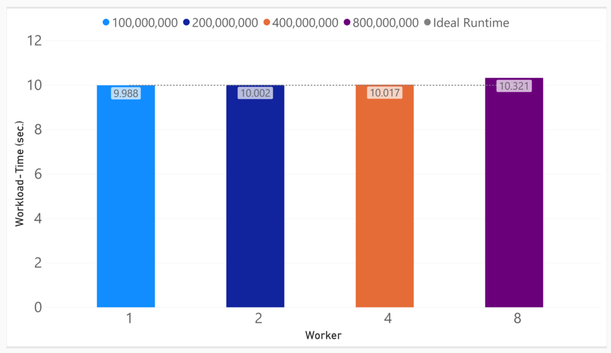

Figure 5:Monte Carlo Scaled Workload

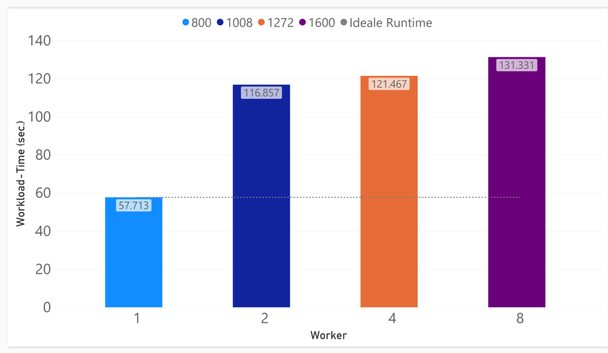

Figure 8:Matrix Multiplication Scaled Workload

HPL Performance: Larger workloads improve the computation-to-communication ratio and therefore parallel efficiency
HPL Strong Scaling: Larger problem sizes utilize additional workers more efficiently; communication prevents ideal linear scaling.
Monte Carlo Strong Scaling: Near-linear scaling for large workloads; efficiency at 8 workers increases from **75% (10M)** to **96% (100M)** and **99% (1B)**.
Matrix Multiplication Strong Scaling: Scales less efficiently, potentially due to increasing communication, memory, and synchronization overhead.
Scaled Workload: Monte Carlo maintains nearly constant runtime as workload grows, whereas the increasing runtime of Matrix Multiplication **may be caused by communication, memory, and synchronization overhead**.

References

[1] M. Morelle. Cloud computing. `https://github.com/MikeMorelle/MikeMorelle.github.io`, 2026.