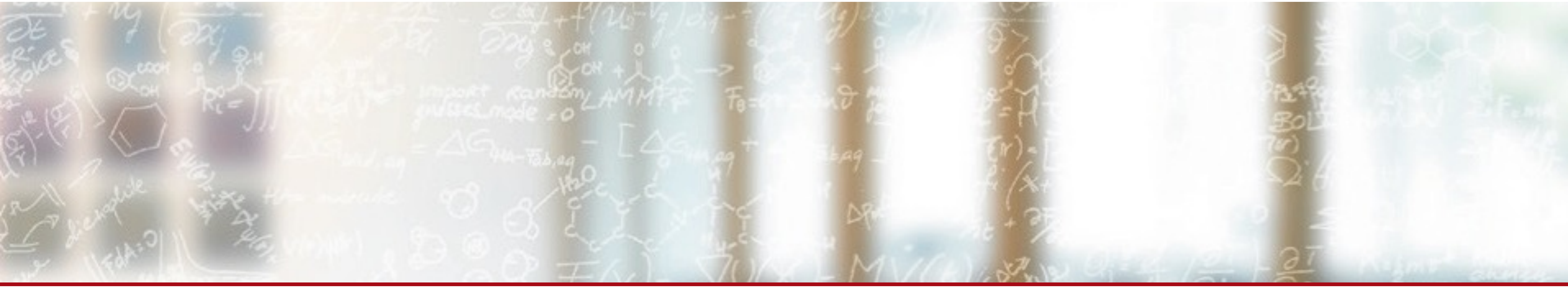
CSCS
Centro Svizzero di Calcolo Scientifico
Swiss National Supercomputing Centre

ETH zürich



EMOI: CSCS Extensible Monitoring and Observability Infrastructure

Massimo Benini Tuesday, May 07th, 2024

Data Warehouse and Data Intelligence - CSCS

CUG 2024 Perth WA

Agenda

Infrastructure

- Background and Motivations
- Components of an OC
- Dynamic deployments of OC
- Hyperconverged K8s infrastructure
- Git-ops with ArgoCD
- Data streams and data mirroring
- Integration with CSA-SMA

Energy dataset

- Slurm and telemetry correlation
- Identify total node energy
- Comparing telemetry energy data Vs Slurm energy data
- GH cabinet power measurements

CSCS

Centro Svizzero di Calcolo Scientifico

Swiss National Supercomputing Centre

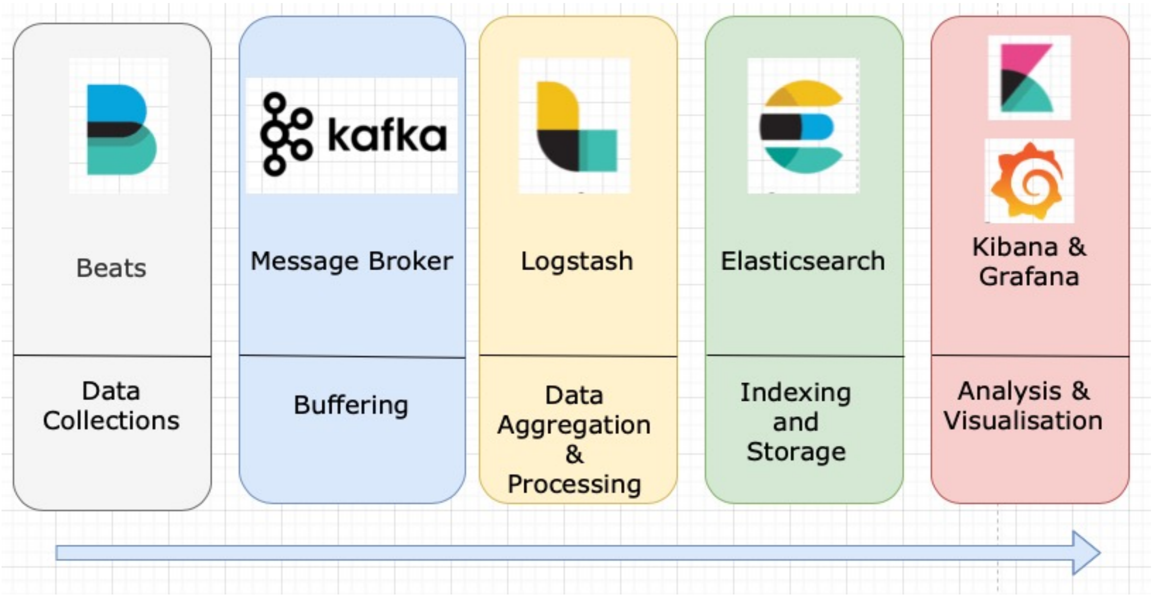
ETH zürich

Developing an Automated Observability System for HPC

Background and Motivations

- The new **Alps** infrastructure introduced the need to significantly scale up our observability platform.
- Managing vast amount of data produced in modern supercomputing, from HW sensors to application data is challenging.
- HW heterogeneity (AMD Rome CPUs, AMD Mi250x, AMD Mi300 GPUs, NVIDIA A100 GPUs and Nvidia GH200) has to be handled properly.
- Full integration of our observability platform (Sole) with the one shipped from HPE-Cray (SMA).
- Flexibility and automation whenever is possible are keys for achieving our goals.
- Streamline the deployment of services: optimize resource utilization and increase operational efficiency
- Embrace a multi-tenancy paradigm for observability platforms.

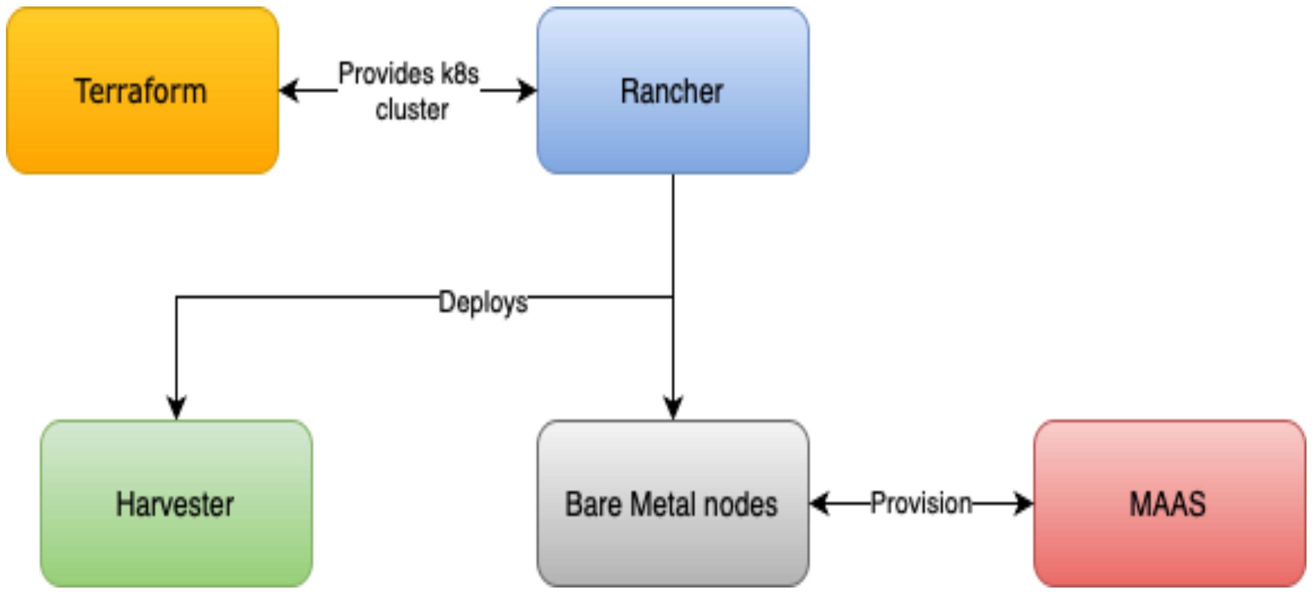
Components of an Observability Cluster



- *Beats*: lightweight data shippers
- *Kafka*: message broker, push model, implements streaming telemetry and acts as a buffer
- *Logstash*: data processing pipeline
- *Elasticsearch*: distributed search and analytics engine designed for storing large volumes of data
- *Kibana & Grafana*: visualization tools, build dashboards, view and analyze data

Dynamic deployments of OC

- *Flexibility*: Multiple physical or virtual Kubernetes cluster dynamically deployed to accommodate custom workflows or external customers
- *Scalability*: provide horizontal scalability to meet changing demands
- Automation: apply IaaC principles and git-ops approach



Hyper converge k8s infrastructure with Terraform - Rancher - Harvester

Terraform

Terraform is an open-source tool for building, changing, and versioning infrastructure safely and efficiently

It allows you to define your infrastructure in a **declarative configuration** language called HashiCorp Configuration Language (HCL)

It supports multiple cloud providers as well as on-premises infrastructure

Terraform performs **idempotent operations**, meaning it only makes necessary changes to achieve the desired state, reducing the risk of unintended changes

It **facilitates collaboration** among teams by allowing them to work on infrastructure changes collaboratively and apply changes using version control systems like Git

Rancher

Rancher is an open-source container management platform that simplifies the deployment and management of Kubernetes clusters.

Rancher allows users to centrally manage **multiple Kubernetes clusters**, regardless of their location or provider, from a single platform.

Rancher offers tools for simple **provisioning and scaling** of clusters, node management, and upgrades

Rancher has strong **security features**, including role-based access control (RBAC), network policies, and integration with identity providers to enhance security and compliance

Harvester

Harvester is an open-source hyperconverged infrastructure solution

Harvester uses **Kubernetes** as its orchestration engine, allowing for effective management of resources and workloads

Built-in virtualization capabilities that enable the creation and management of virtual machines (VMs) directly within it using kubevirt

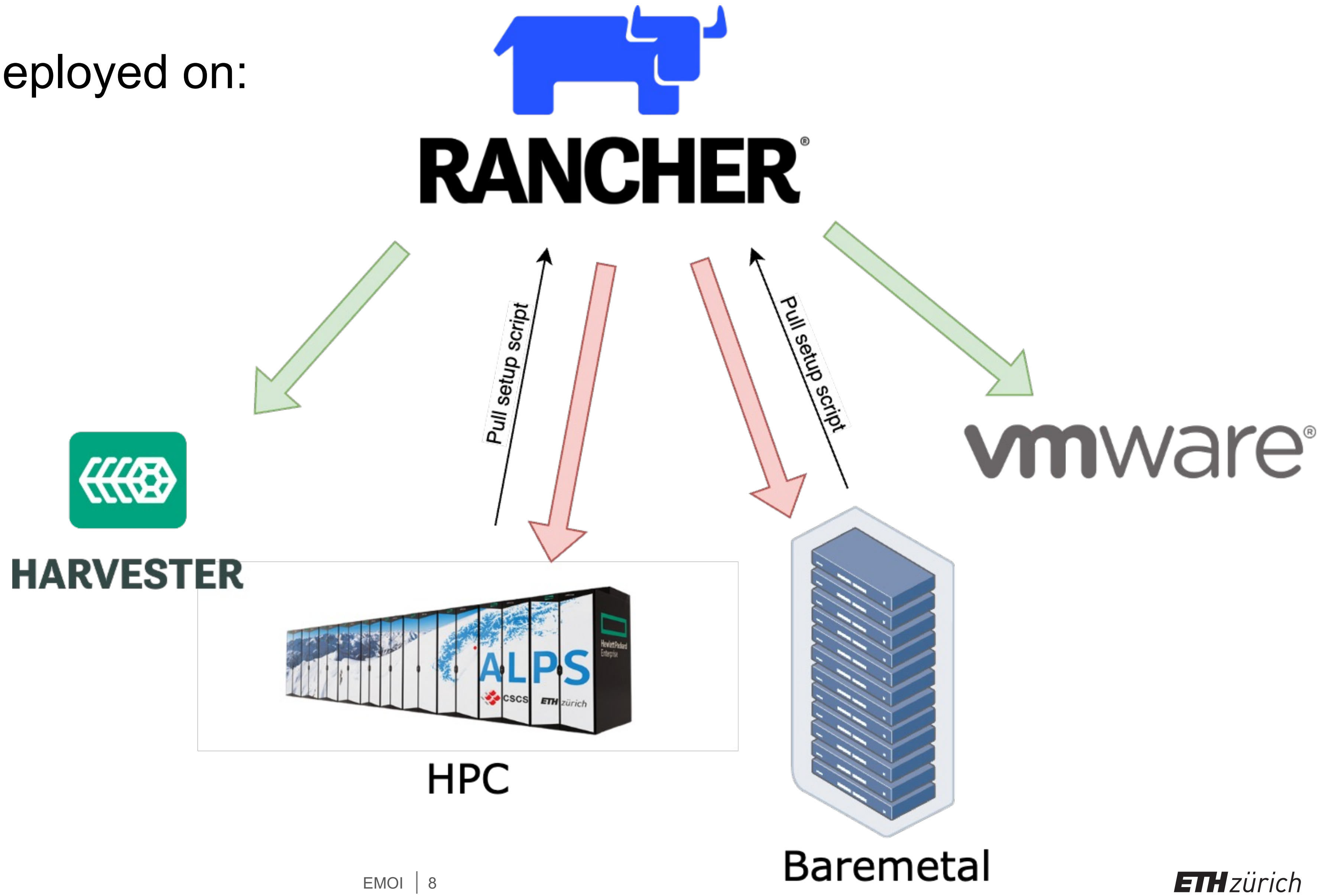
Storage management through Longhorn (distributed block storage for Kubernetes)

Networking integration ensures reliable communication between virtual machines (VMs) and external services while maintaining isolation across multiple VLANs

Workflow

Kubernetes clusters deployed on:

- Harvester
- VMware
- Baremetal
- HPC (CSCS Alps)

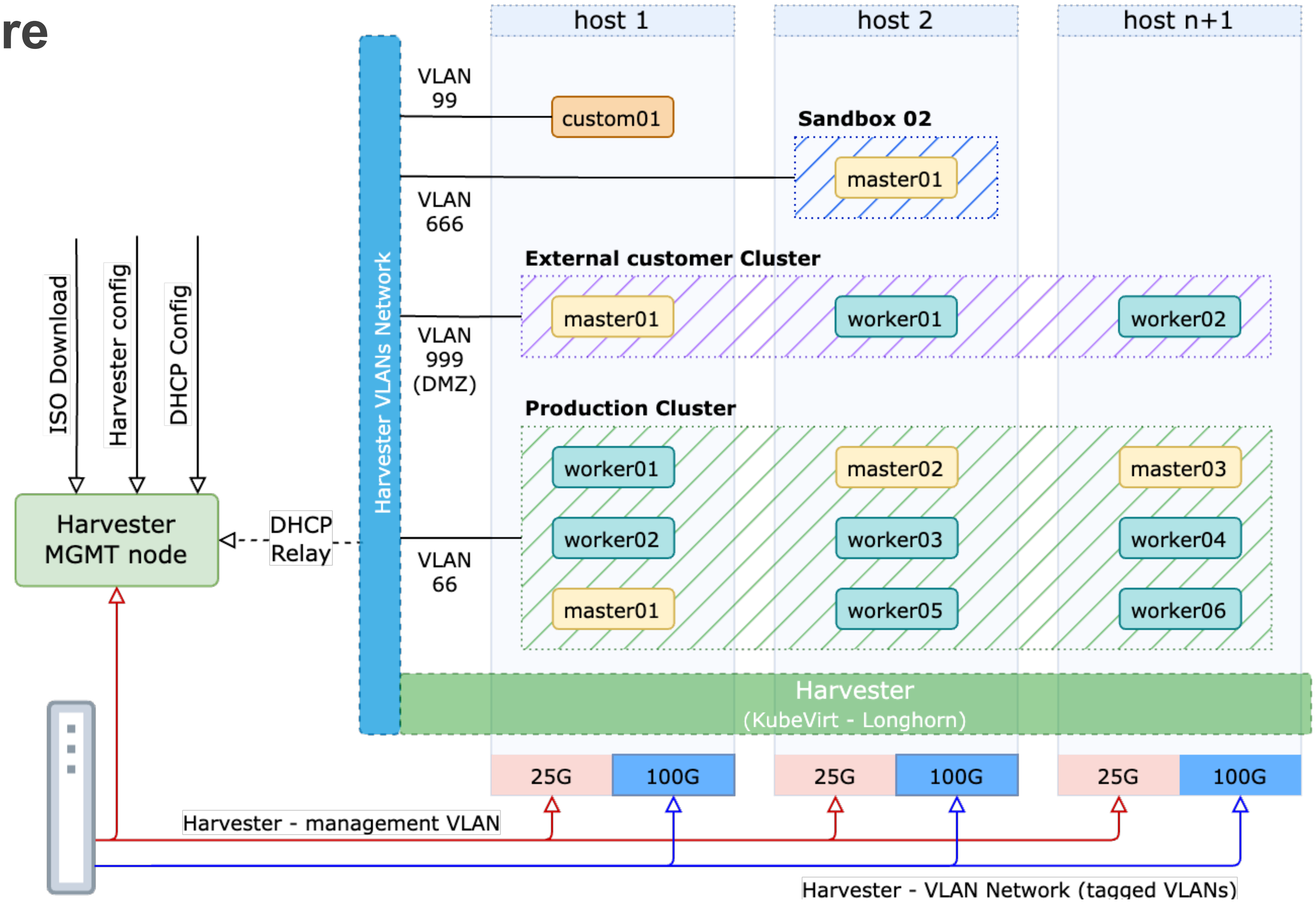


Harvester architecture

- **Management node:**
  - DHCP Relay for VMs
  - Harvester host setup
    - iPXE Boot
    - ISO Repository
    - Hosts config

- **Harvester host:**
  - MGMT VLAN (25G)
    - live migration
    - Kubernetes OPS
    - Harvester StorageClass
  - tagged VLANs (100G)
    - All cluster workflow

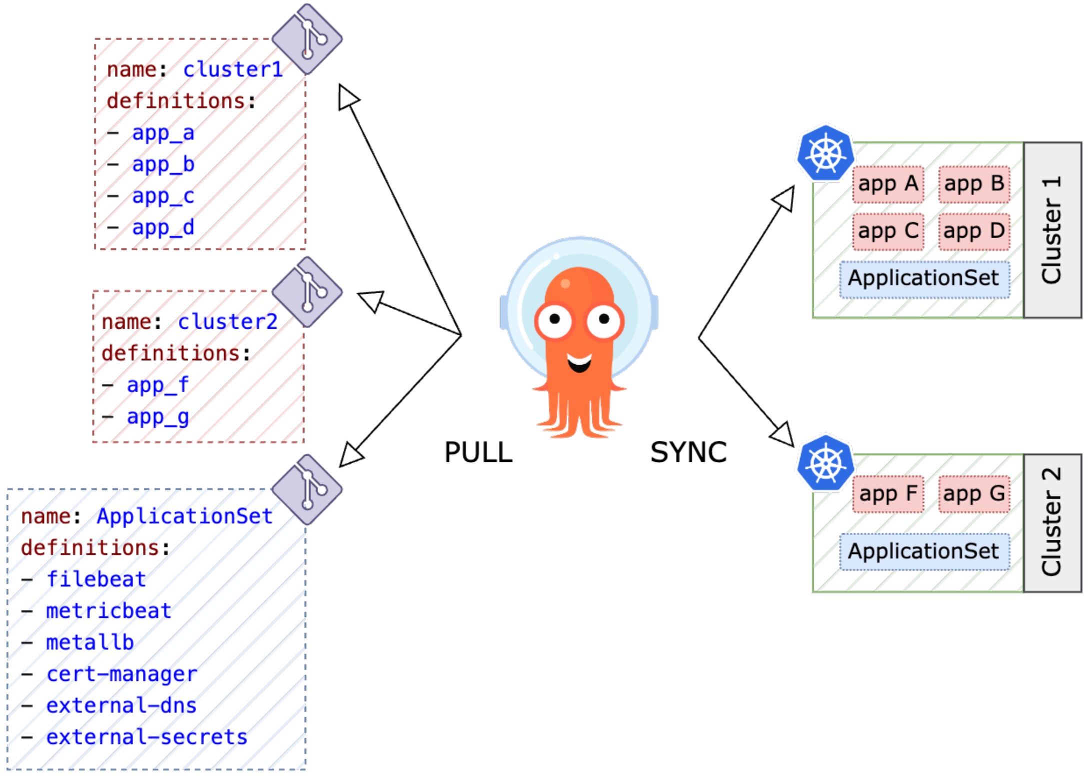
- **Storage:**
  - Local NVMEs
  - Ceph



Git-ops with ArgoCD

Argo CD is an open-source continuous delivery tool specifically designed for Kubernetes that follows the GitOps methodology.

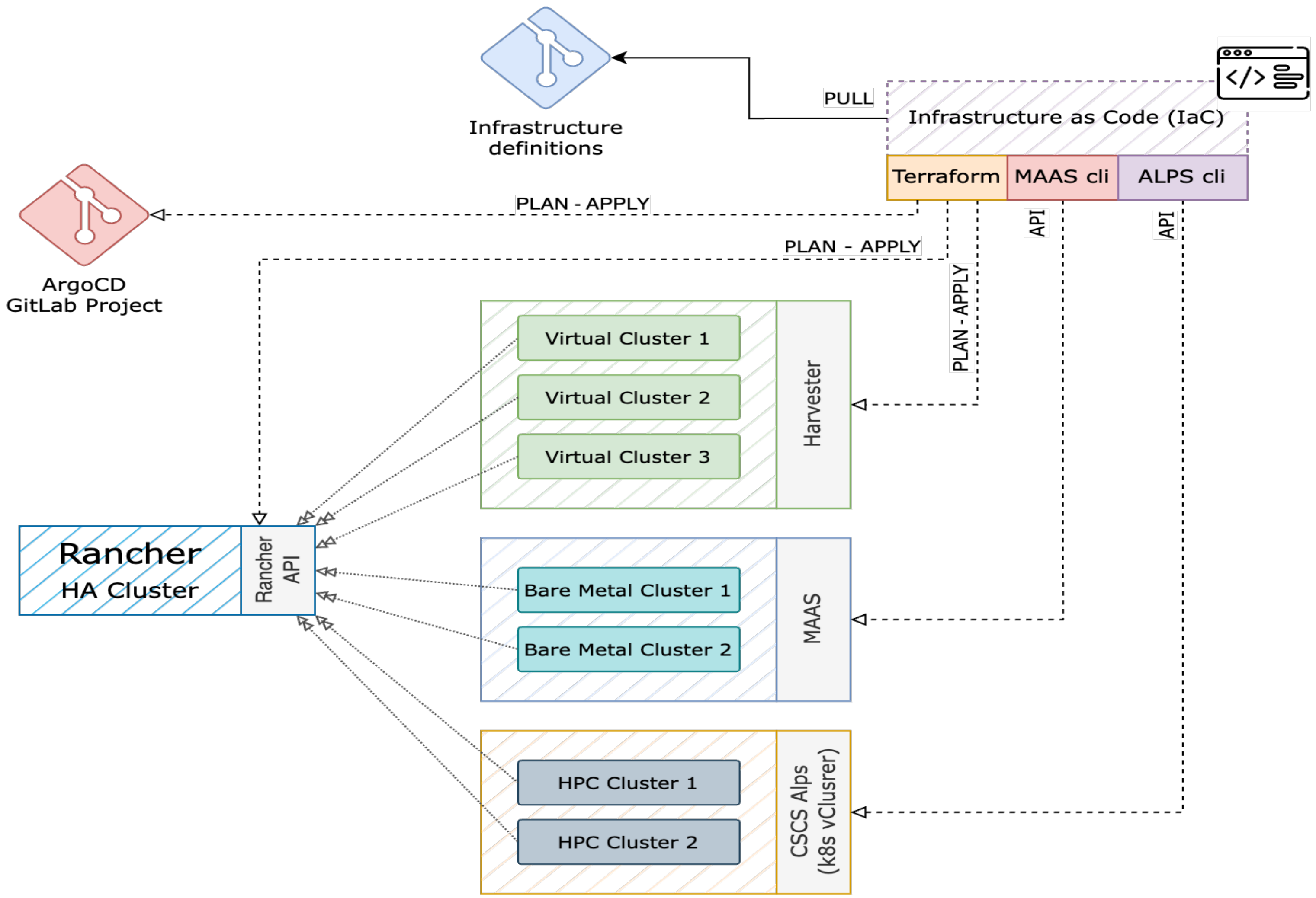
- **Declarative Configuration**: Users define application deployments declaratively using Kubernetes manifests or Helm charts stored in Git repositories. Argo CD then ensures that the actual cluster state matches the desired configuration
- **Graphical UI**: Argo CD provides a user-friendly web interface for visualizing and managing application deployments. Additionally, it offers a command-line interface (CLI) for scripting and automation.
- Application definitions: for each cluster a separate git repo with all apps manifests
- ApplicationSet: Applications deployed on all clusters



CSCS

ETH zürich

General overview



- **Harvester (or VMware)**

  `$ terraform apply`
  - Cluster definition
  - Triggers the VMs pools creation
  - Rancer will runs the cluster join command on the newly created VMs

- **MAAS (Bare Metal nodes)**

  `$ terraform apply`
  - Cluster definition

  `$ ansible-playbook deploy-rke2.yml`
  - Nodes will join the cluster

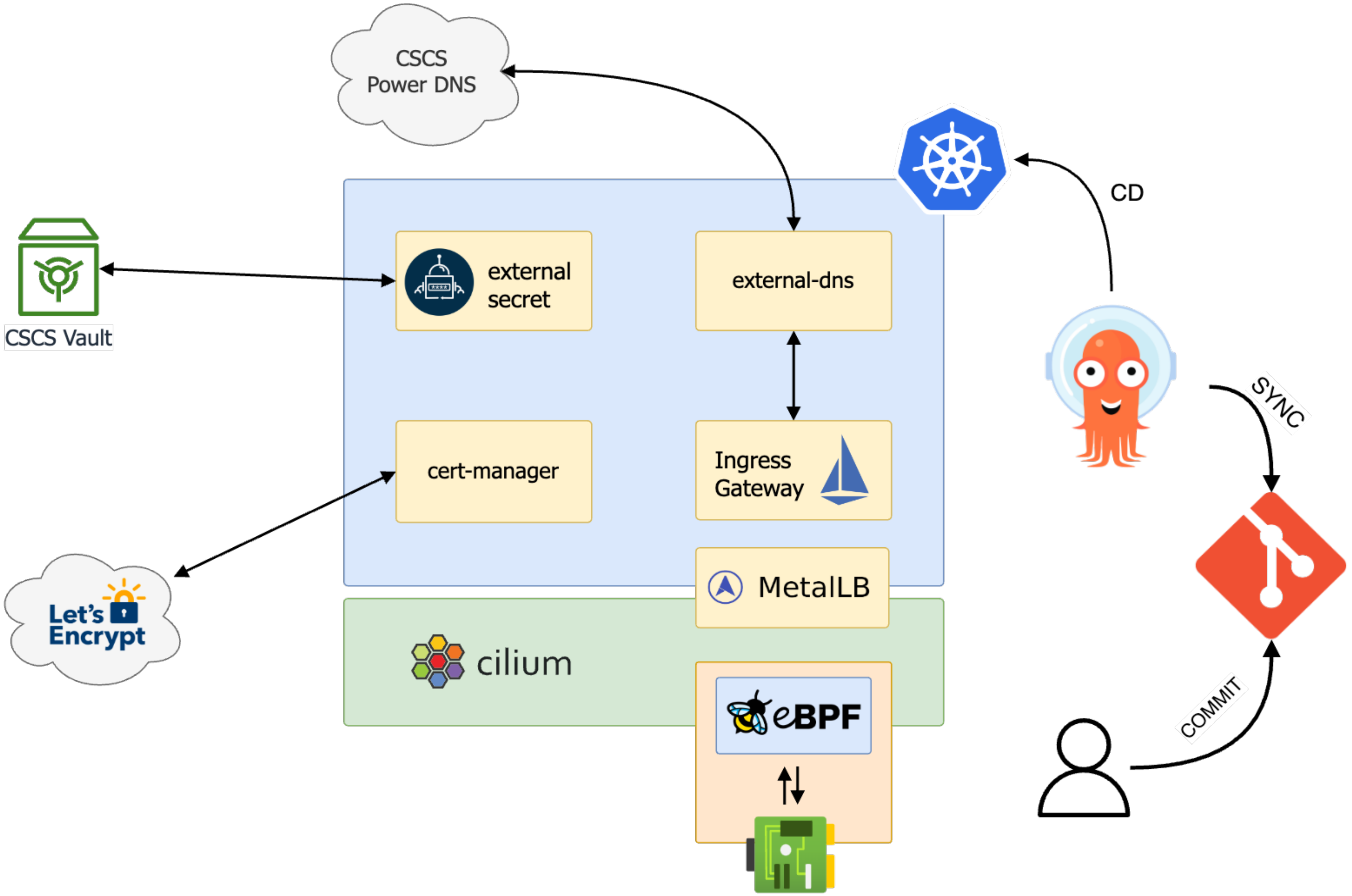
- **CSCS Alps**

  `$ terraform apply`
  - cluster definition
  - Master nodes creation on Harvester

  `$ ./rancher-agent-install.sh`
  - Worker nodes will join the cluster ()

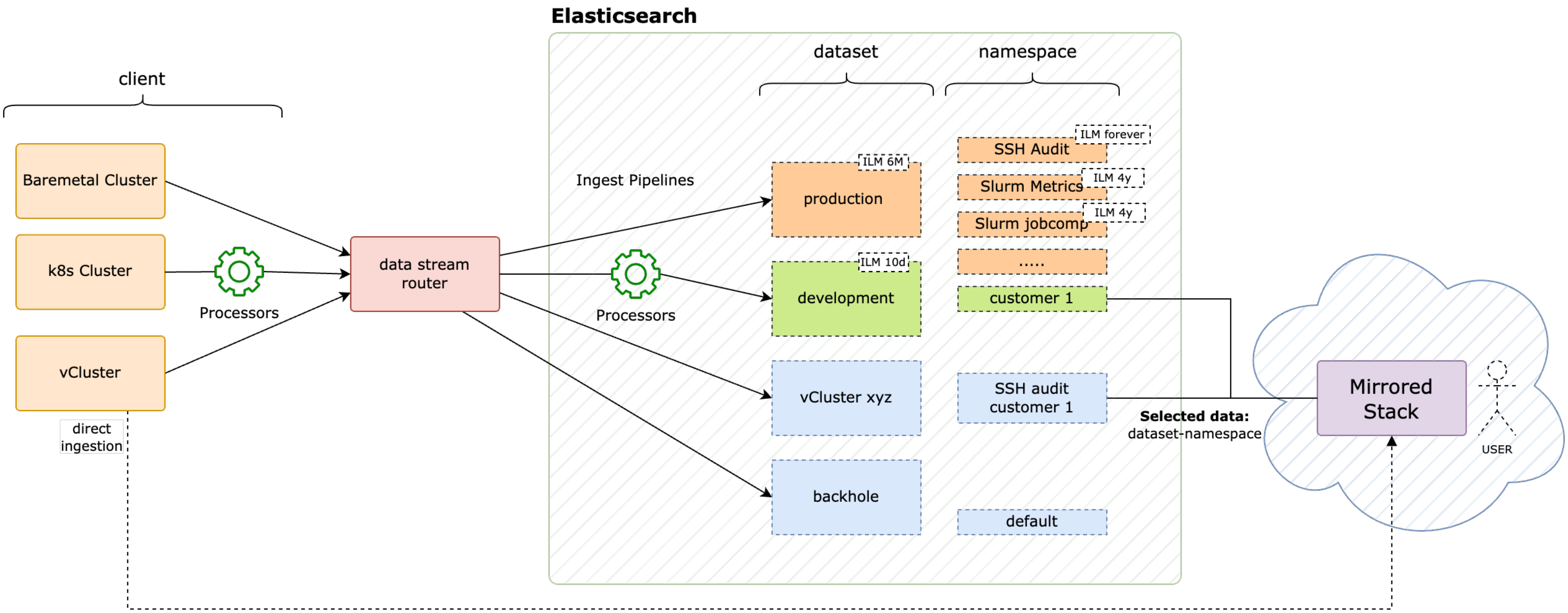
Central observability cluster: base configuration

- CNI: Cilium
  - Service mesh (eBPF)
  - Hubble (observability UI)
- Istio (only Ingress GWs)
  - No sidecars
  - Testing new API Gateway
- MetalLB
  - Currently still via ARP
  - BGP in the future
- Automated DNS and Certs
- GitOps manages:
  - Applications
  - Base components
- External Secret

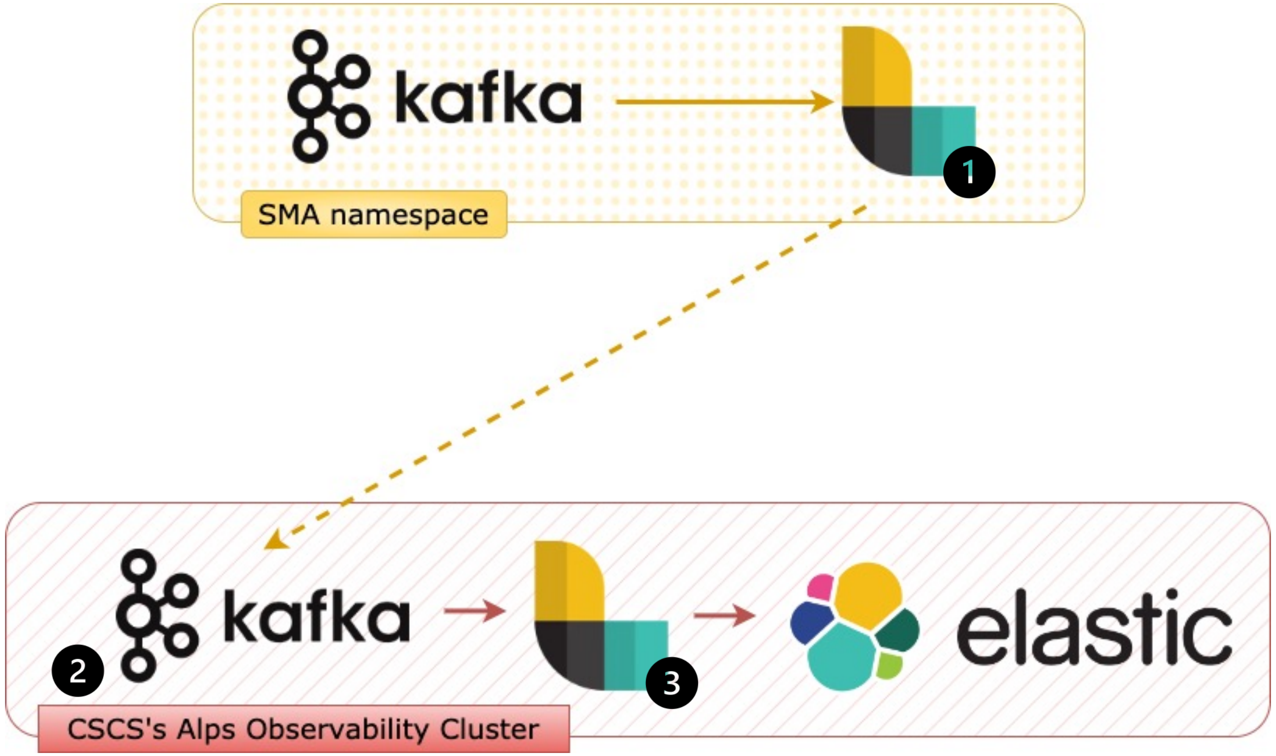


Data Streams workflow and data mirroring

Datastreams are a simplified routing tecnique with an index names abstraction



Integratation with CSM-SMA Kafka Bus



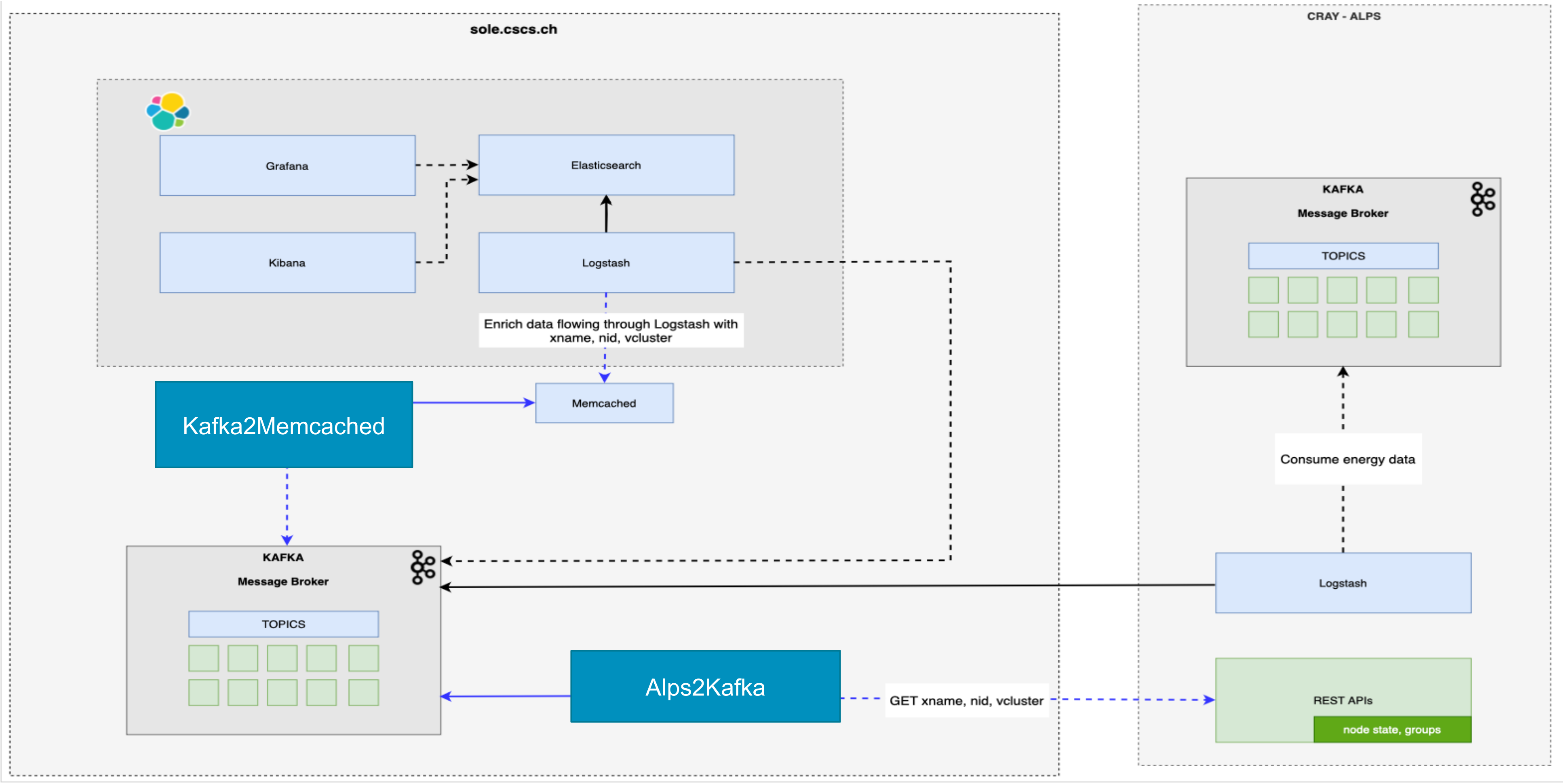
1. Selectively choose which topic to mirror:

Domains	Topics
System nodes	cray-node
Fabric Telemetry	cray-fabric-telemetry
Power, Energy and Voltage	cray-telemetry-energy cray-telemetry-voltage cray-telemetry-power
Environmental Telemetry	cray-telemetry-temperature cray-telemetry-fan cray-telemetry-pressure
System Hardware	cray-dmtf-resource-event cray-hsmstatechange-notifications
Kubernetes	cray-logs-containers

2. Split message bundles (ex. per sensor)

3. Further manipulate and enrich data

Enriching the data: Alps2Kafka and Kafka2Memcached



CSCS
Centro Svizzero di Calcolo Scientifico
Swiss National Supercomputing Centre

ETH zürich

Integrating Operational and Energy Dataset

Energy dataset: SLURM and Telemetry correlation

GOALS

Node – level

For the 4 following nodes:

1. nid001001 - eiger for multicore processor with 2 CPU : 0 GPU
2. nid002556 - clariden for amdgpu processor with 1 CPU :4 GPU
3. nid002792 - clariden for nvidia gpu processor with 1 CPU :4 GPU
4. nid001804 - santis for grace-hopper processor with 4CPU :4GPU

- 1 . Identify which component of the telemetry corresponds to the total energy of the node.
- 2. Compare telemetry energy data of the node with Slurm energy data of the jobs

Cabinet –level

- 3. cabinet grace-hopper ICON tests - verify power, temperature and current telemetry data

1. Identify which telemetry component corresponds to the total energy of the node

- Telemetry - energy:

  $$E_{telemetry} = E_{telemetryJobEnd} - E_{telemetryJobStart}$$

  frequency of measurement: around 1 Hz

- PM file – energy (4 components: CPU, Memory,GPU,Total):

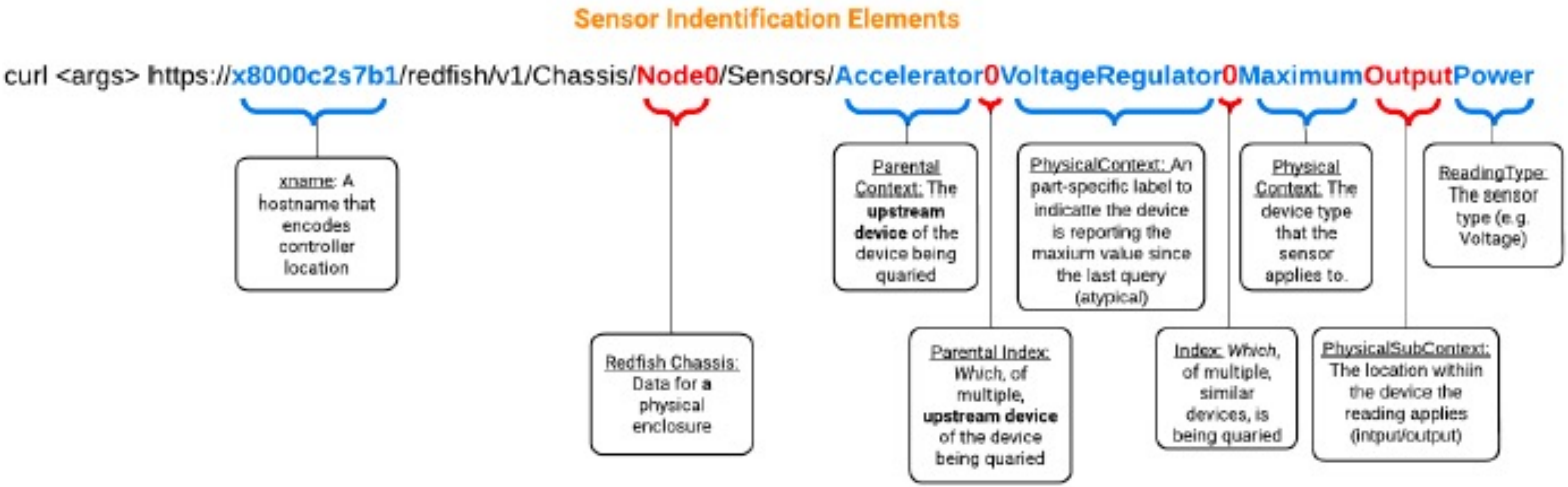
  $$E_{componentx\_pmfile} = E_{componentx\_pmfileJobEnd} - E_{componentx\_pmfileJobStart}$$

  Frequency of measurement: around 10 Hz

- SLURM - energy: retrieves the job-total-energy from the pm-file total energy

  $$E_{SLURM} = E_{tot\_pmfileJobEnd} - E_{tot\_pmfileJobStart}$$

Redfish call anatomy:



1. Identify which telemetry component corresponds to the total energy of the node

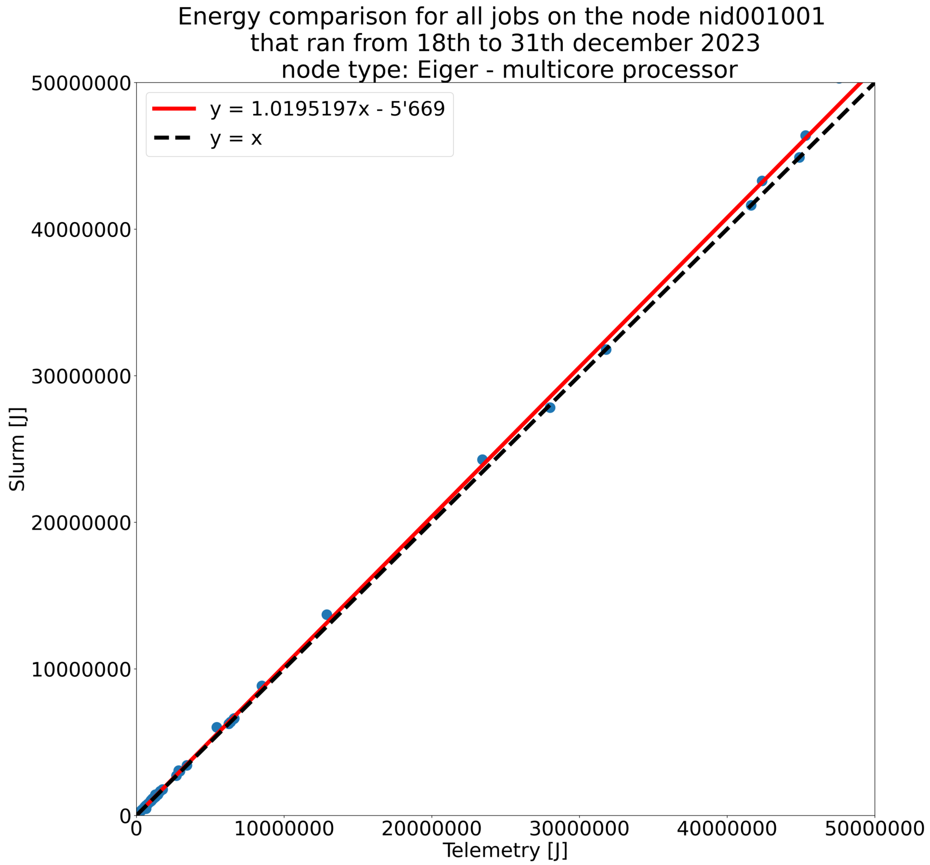
- Telemetry Data query:

$$Energy_{node} = Sensor.Location :< nodex >$$
$$\&MessageId : CrayTelemetry.Energy$$
$$\&Sensor.ParentalContext : Chassis$$
$$\&Sensor.PhysicalContext : VoltageRegulator$$
$$\&Sensor.PhysicalSubContext : Input$$
$$\&Sensor.Index : 0$$

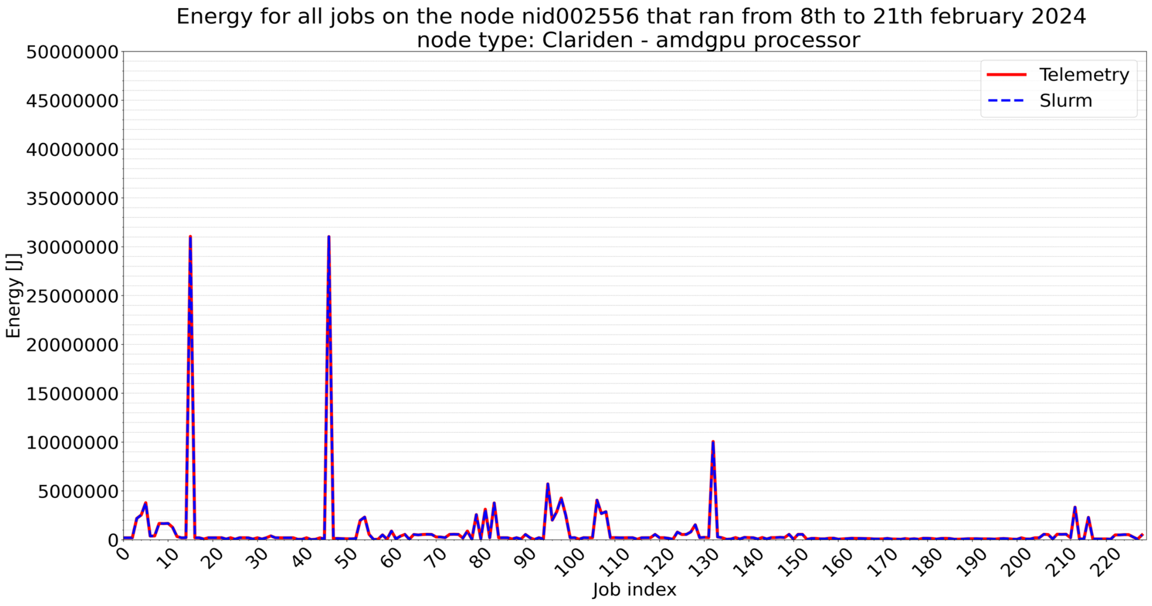
kind	task	start	end	CPU [J]	Memory [J]	VoltageRegulator [J]	node energy [J]
Telemetry	all	2024-01-29T14:17:32	2024-01-29T14:27:01	69'975	73'987	174'706 ≡	174'706
pm file				69'686	72'185		170'259
Telemetry	none	2024-01-29T14:17:38	2024-01-29T14:18:28	2'670	6'011	10'285	10'285
pm file				2'616	6'125		10'370
Telemetry	1CPU cores	2024-01-29T14:18:29	2024-01-29T14:19:29	3'138	7'202	12'661	12'661
pm file				3'172	7'349		12'916

2.Compare telemetry energy data of the node with Slurm energy data of the jobs

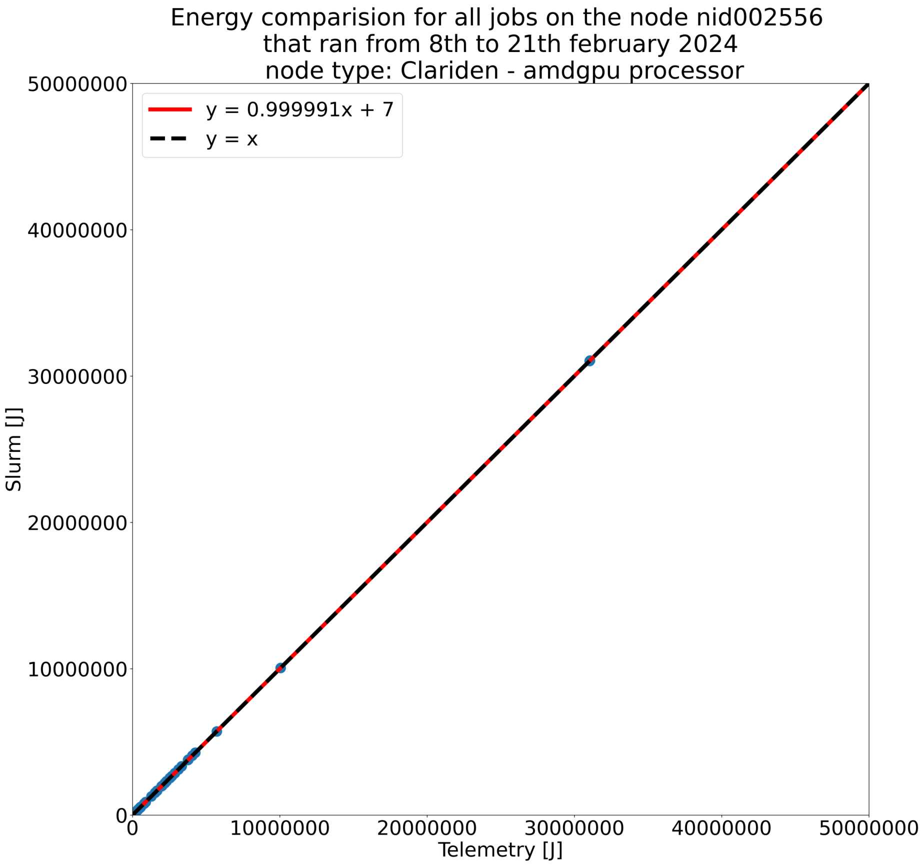
- average of the energy delta is around -0.8%.
- telemetry – SLURM correlation value is 0.9997357



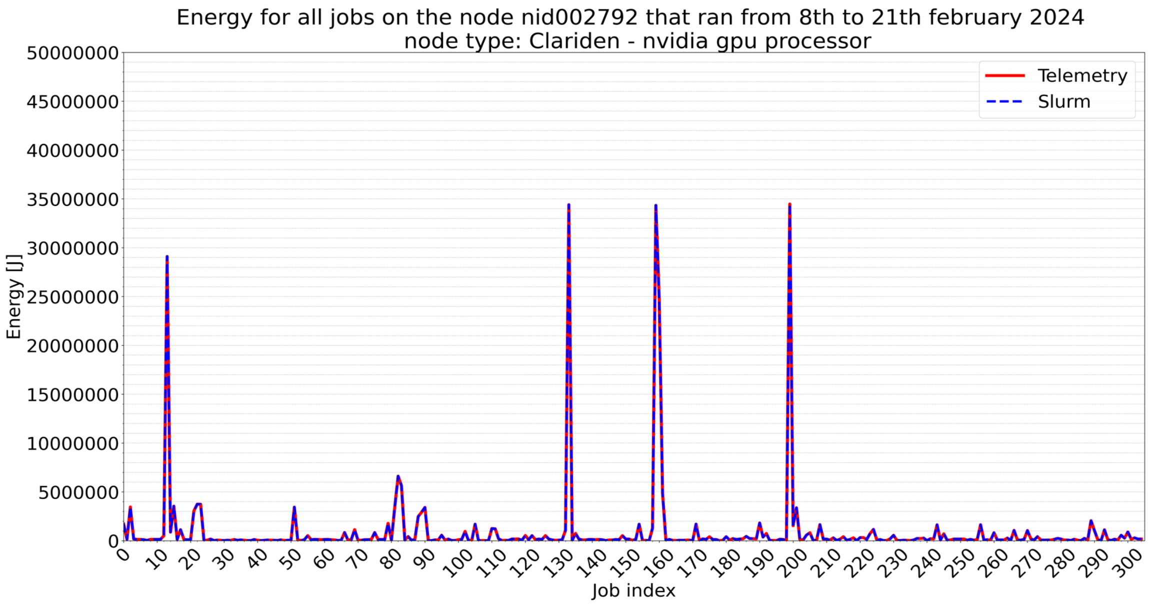
2. Compare telemetry energy data of the node with Slurm energy data of the jobs

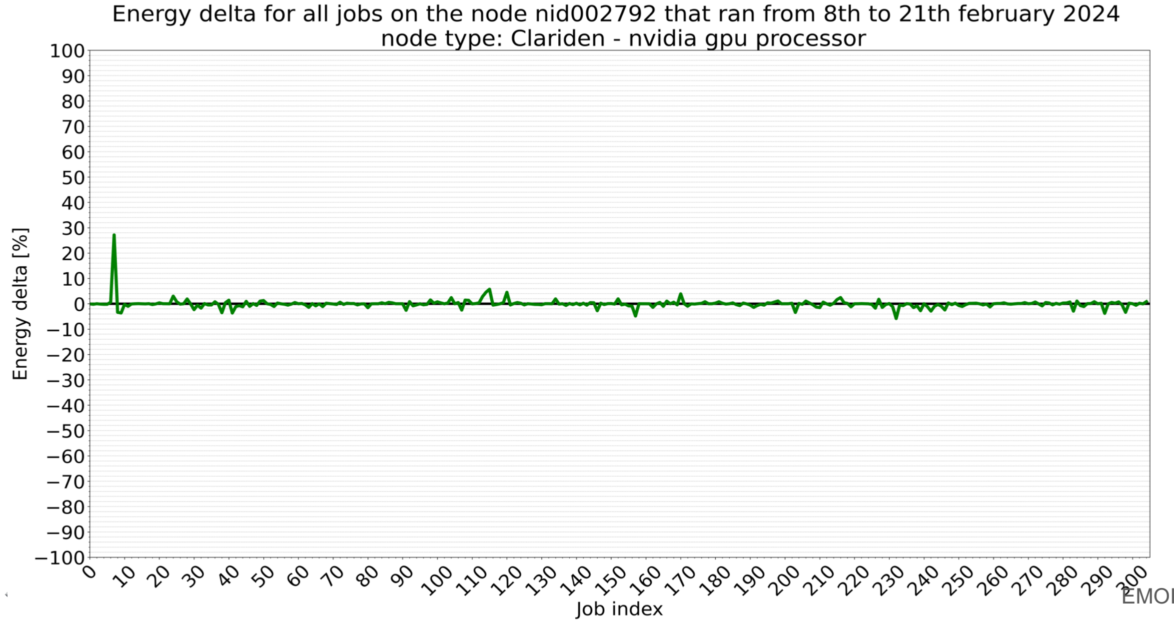


- The average of the energy delta is around -0.14%.
- telemetry – SLURM correlation value is 0.9999999

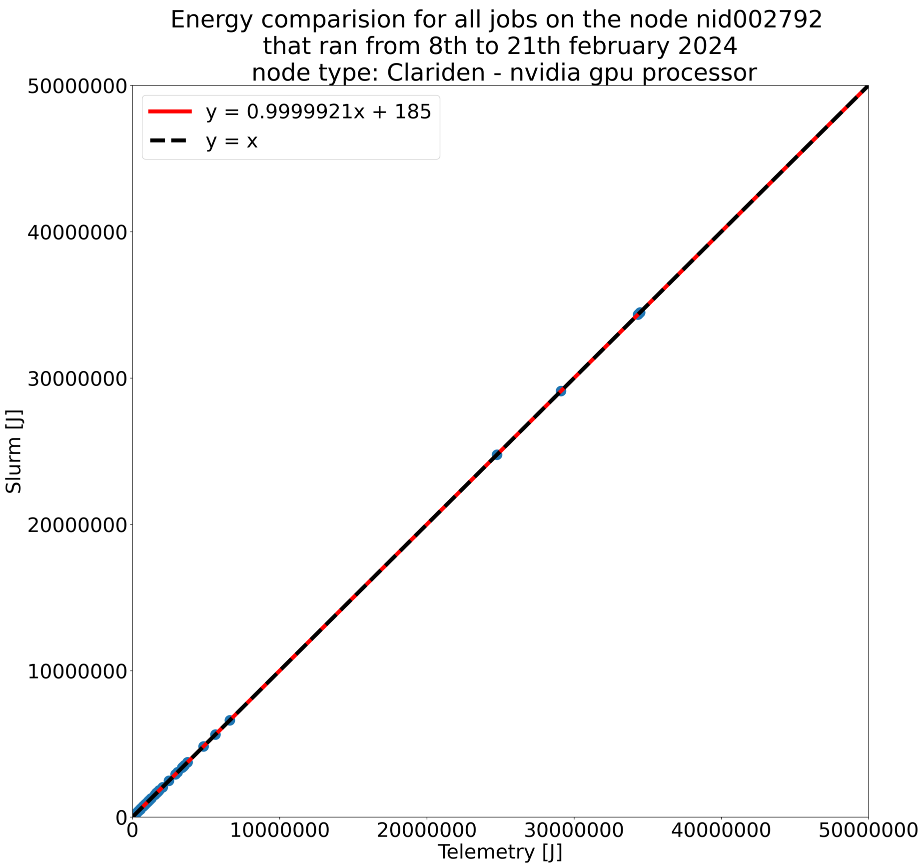


2.Compare telemetry energy data of the node with Slurm energy data of the jobs



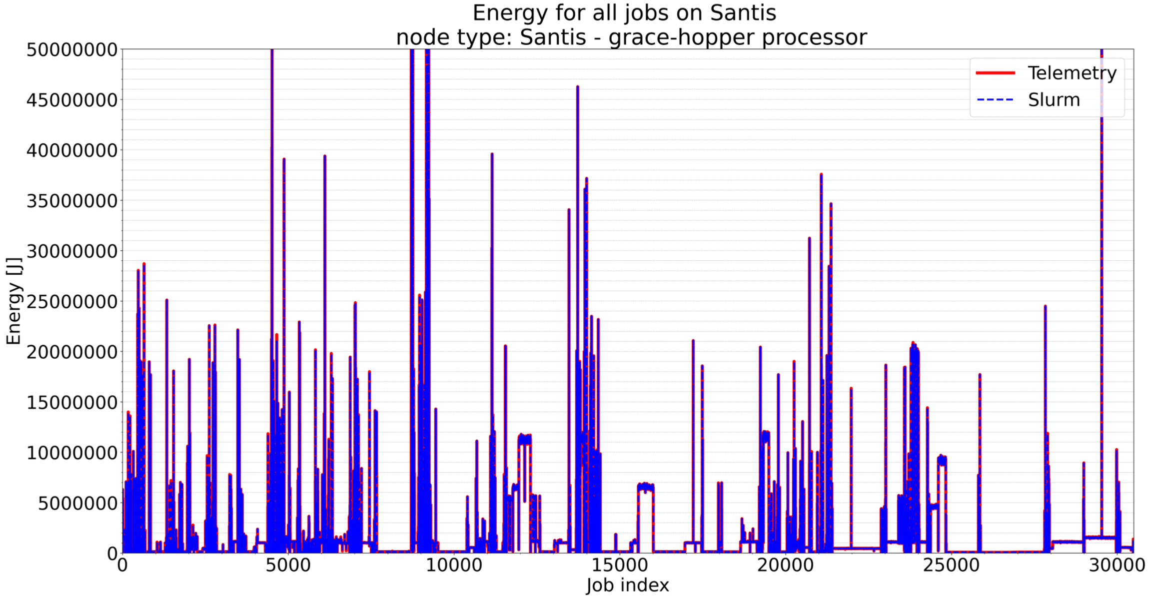


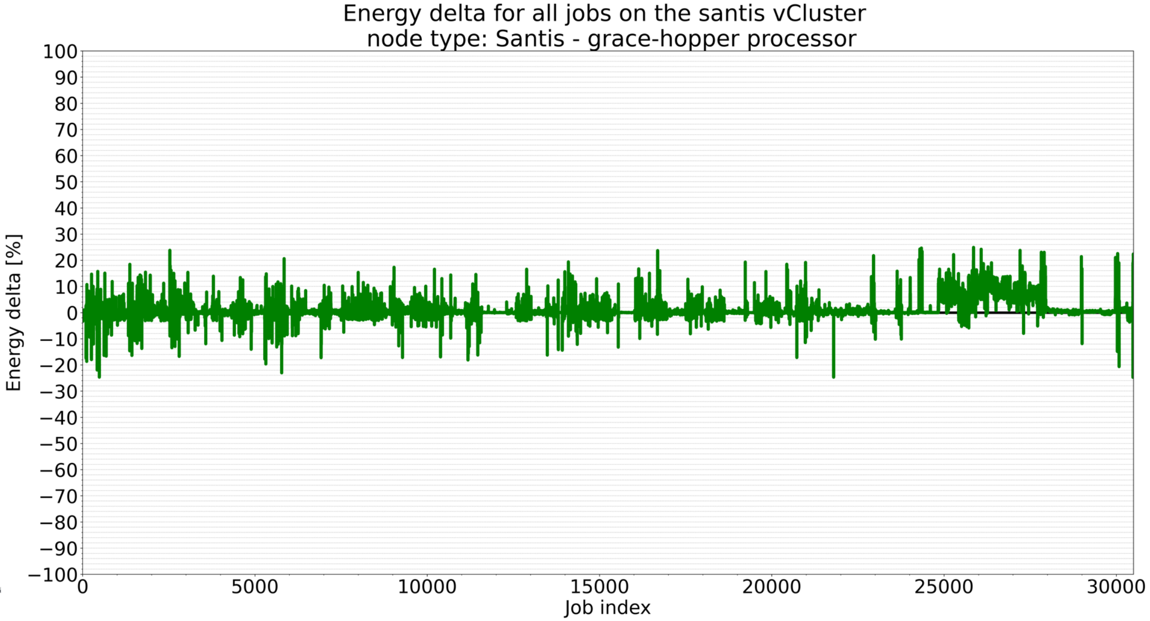
- The average of the energy delta is around -0.04%.
- telemetry – SLURM correlation value is 0.9999997



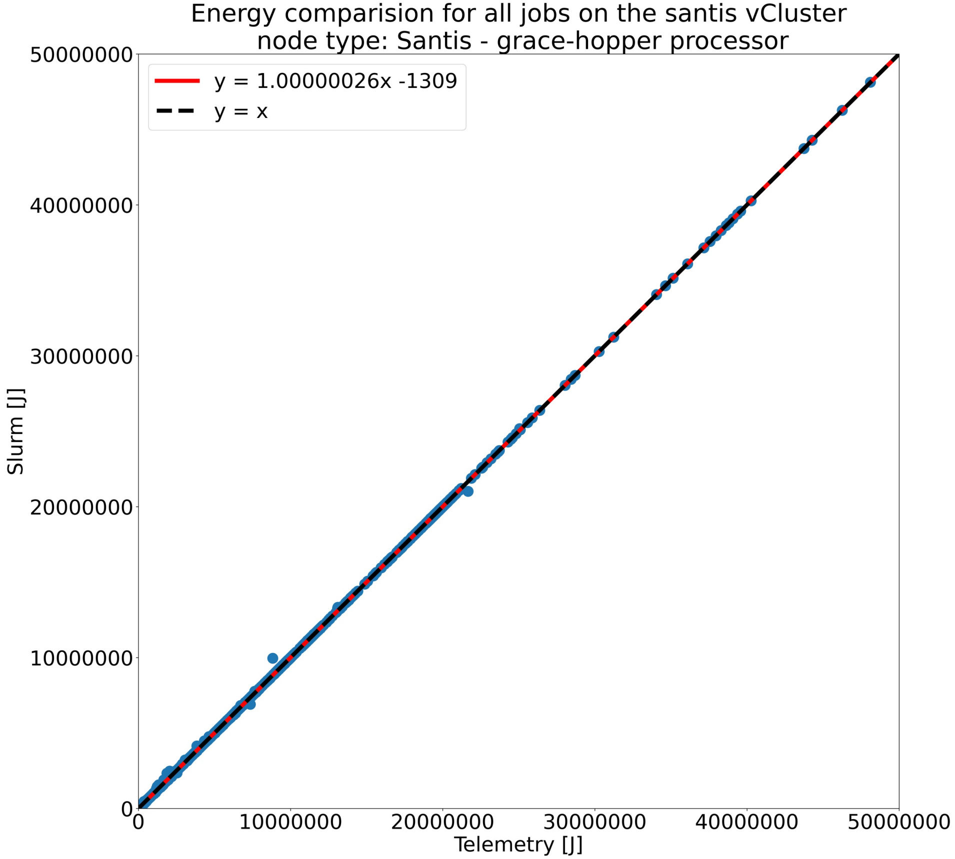
ETH zürich

2. Compare telemetry energy data of the node with Slurm energy data of the jobs

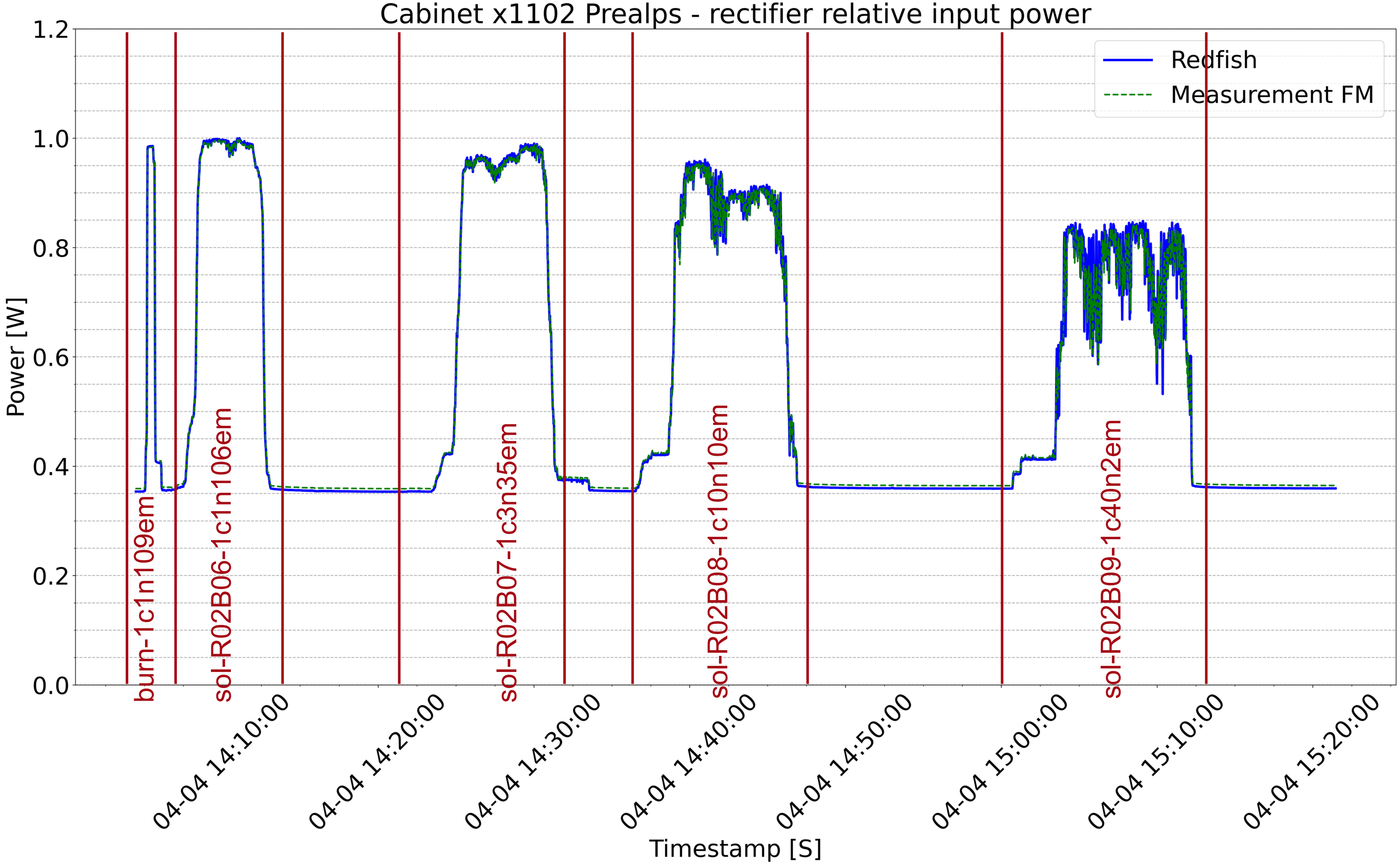




- The average of the energy delta is around 0.978%.
- telemetry – SLURM correlation value is 0.9999957.

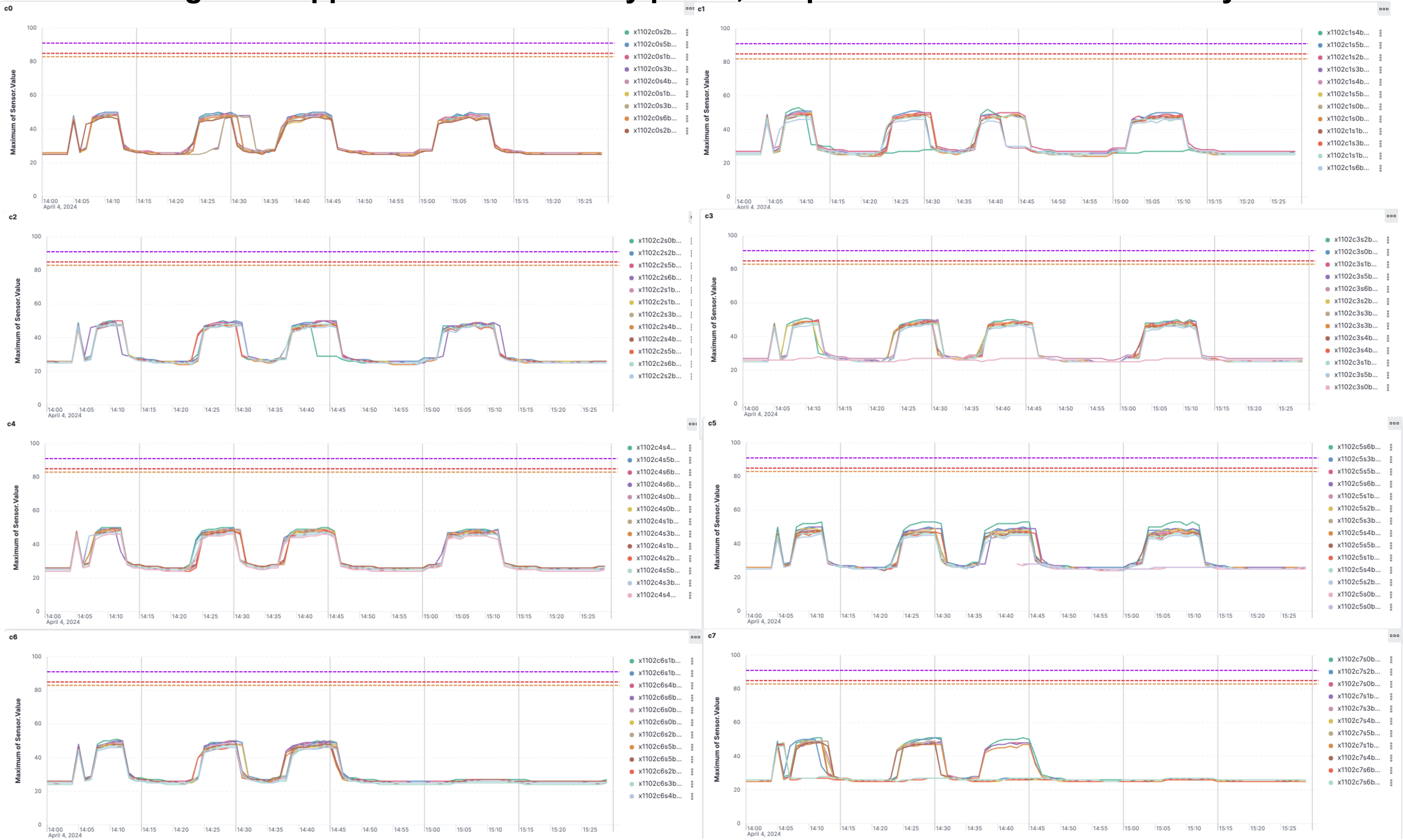


3. cabinet grace-hopper ICON tests - verify power, temperature and current telemetry data



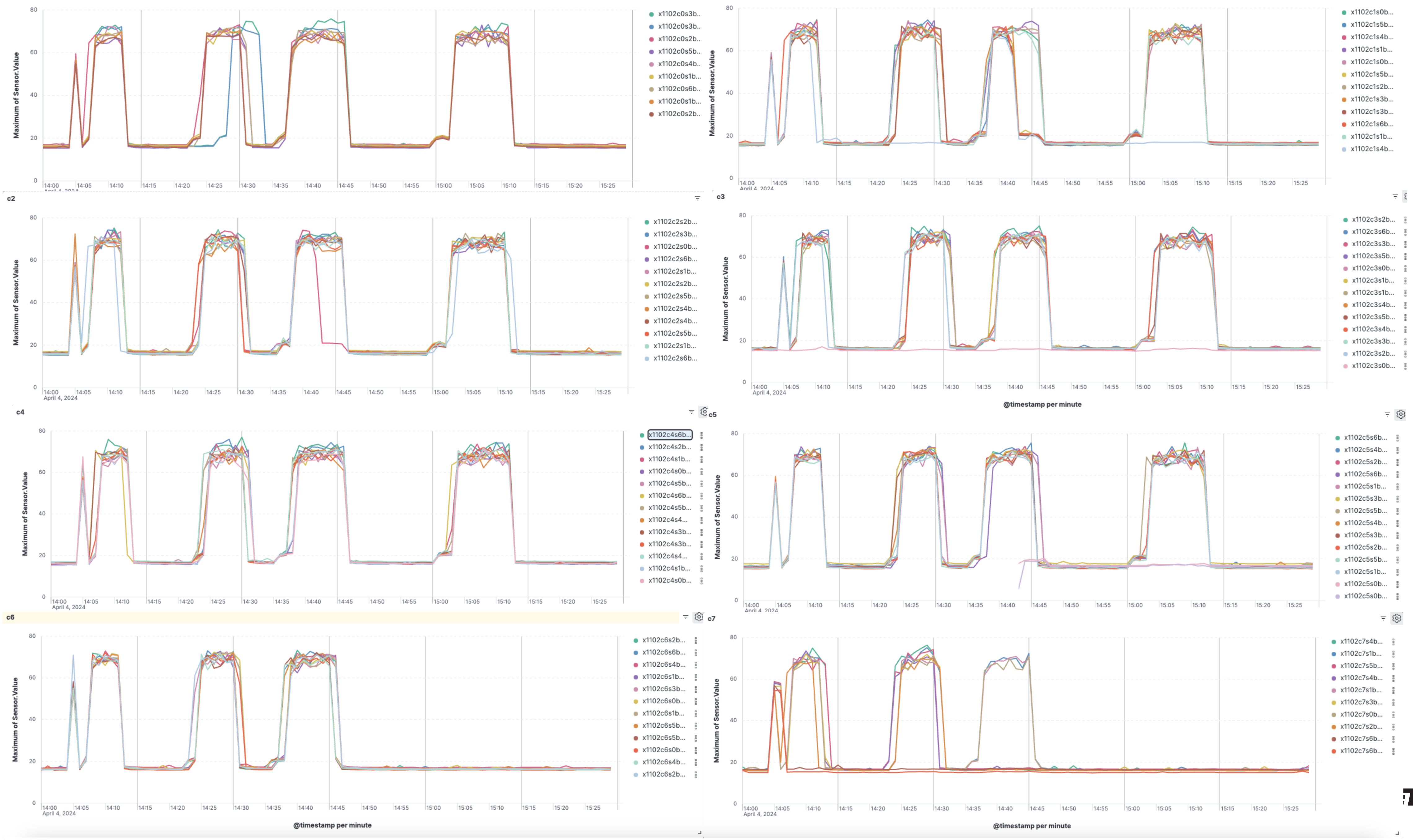
ETH zürich

3. cabinet grace-hopper ICON tests - verify power, temperature and current telemetry data



ETH zürich

3. cabinet grace-hopper ICON tests - verify power, temperature and current telemetry data



Appendix

Measuring instruments (8 units)

Model:	Chauvin Arnoux	PEL103
Calibration date:	06.2023	
Clamp model:	Miniflex MA193	
Precision:	± 0.5%	

Conclusions

- The **integration of SMA into the EMOI** was possible due to its kafka-centric model with very low overhead.
- **Performance tuning** of the various componets, along data pipelines, is key when dealing with massive data ingestion (5Vs: Velocity, Volume, Value, Variety and Veracity). We build several dashboard to help us tuning our components and we are setting up a datapipeline framework with Apache Airflow to detect anomalies.
- Adopting a **git-ops approach is a real advantage**. The flexibility given allow us to easily create and destroy o11y clusters on demand and selectively ingest data.
- We have now enabled energy and power data collection, the next step is to start using these data to **optimize energy consumption.**

CSCS
Centro Svizzero di Calcolo Scientifico
Swiss National Supercomputing Centre

ETH zürich

Thank you for your attention.