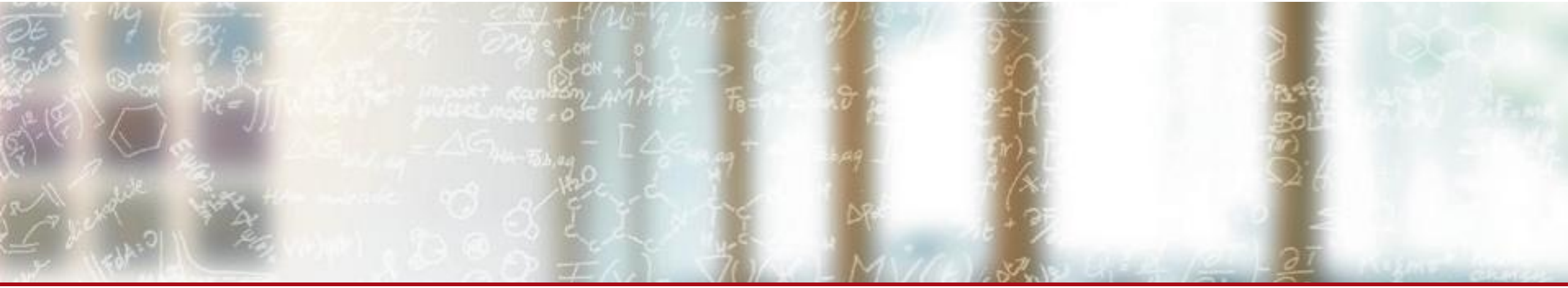


The state of OpenCHAMI at CSCS

HPC AC
Manuel Sopena Ballesteros, CSCS
April 22, 2026

Agenda

1. Introduction to CSCS
2. Management plane (CSM vs OpenCHAMI)
3. The challenge
4. The plan

CSCS
Centro Svizzero di Calcolo Scientifico
Swiss National Supercomputing Centre

ETH zürich

Introduction to CSCS

CSCS Mission

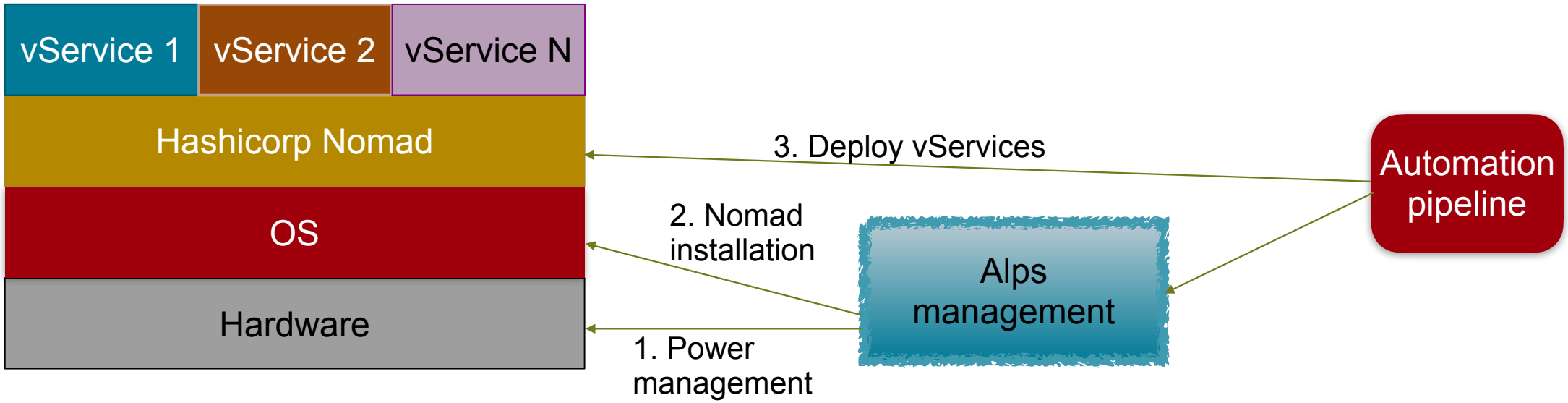
- CSCS mission is to develop and operate a high-performance computing and data research infrastructure that supports world-class science in Switzerland

CSCS organization

- Working structure to manage the infrastructure
- WS in charge of the compute node life cycle
- WS responsible for the storage
- WS for networking
- WS for security
- WS for metrics/logs/alerting
- ...

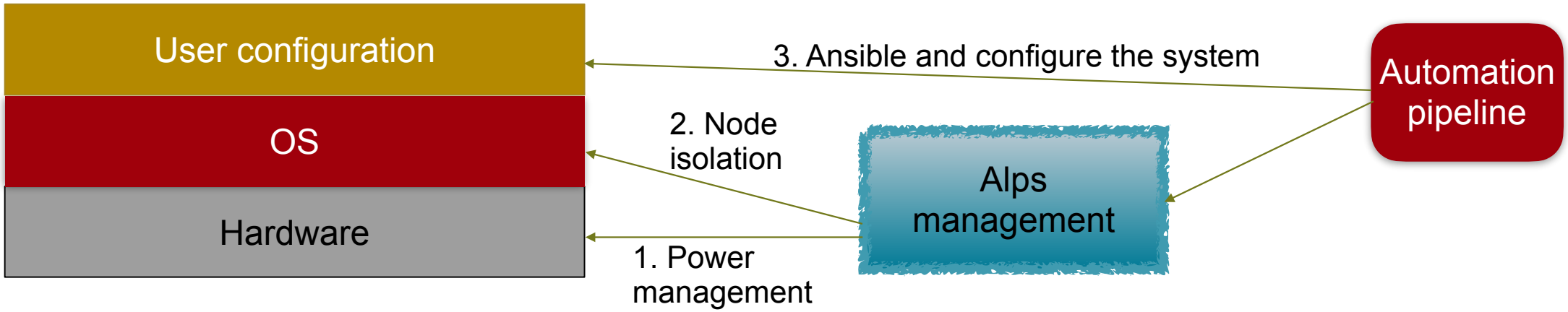
Infrastructure ecosystem (hard tenant)

- Management plane
- Hashicorp Nomad



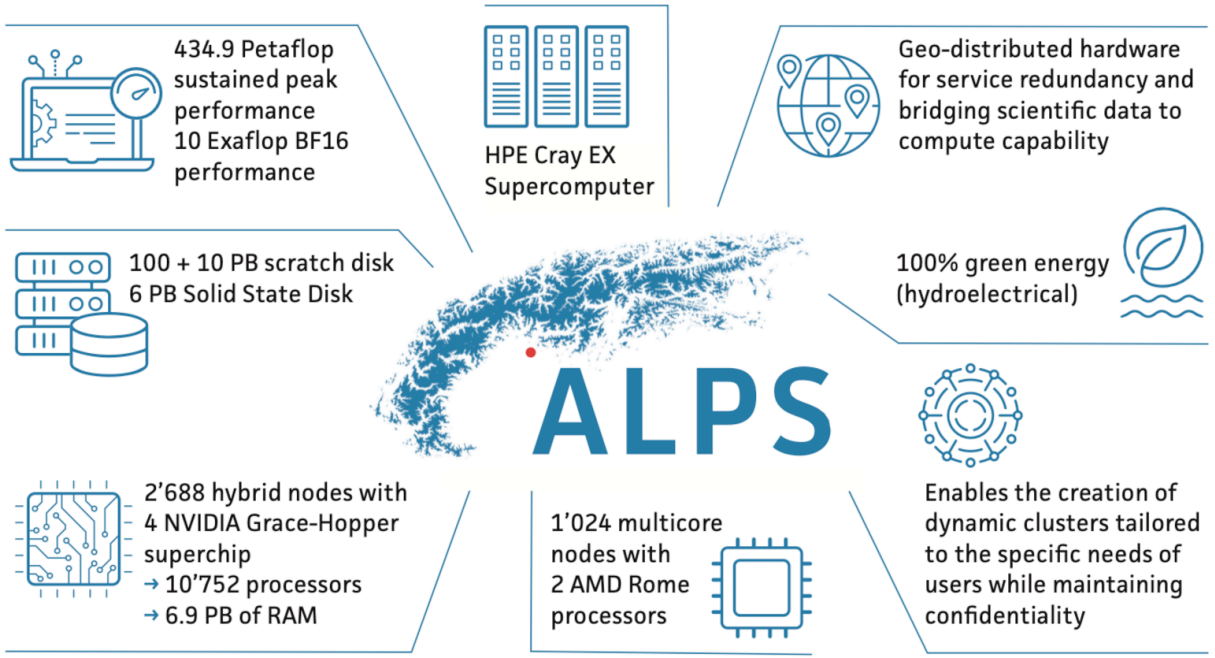
Infrastructure ecosystem (soft tenant)

- Management plane



Alps challenges

- Mix of different hardware
- Geo distribution
- Multi-tenancy w/o virtualization
- Different types of workloads



Management plane (CSM vs OpenCHAMI)

Alps Current State - CSM

- Current management plane
- Provided by HPE Cray
- Used by CSCS since ~2021
- Batteries included
- Micro services architecture
- "Cloud" interfaces
- Based on kubernetes

CSM features

- Image build pipeline
- Data defined cluster
- Power management
- Hardware inventory
- Console management
- Diagnostics
- WLM included
- Runtime configuration pipeline (Ansible)
- Monitoring/alerting
- Identity Access Manager (IAM) - Keycloak
- Software attestation

CSM challenges

- Complex
- Management and Compute highly coupled
- Incidents in management plane affecting compute nodes
- Project continuity
- Vendor locked (can't use with other hardware)

OpenCHAMI

- Management plane
- HPSF (Linux Foundation Project)
- Members (CSCS, LANL, NERSC, Bristol, HPE, Dell)
- Microservice architecture
- Community driven
- "Cloud" interfaces
- Vendor neutral
- Composable
- Based on CSM (subset of features)
- Simplicity (less moving parts)
- Unopinionated on how to install

OpenCHAMI features

- Boot management
- Redfish integration*
- Hardware inventory *
- Power management *
- Node inventory
- Image builder
- Runtime configuration (Cloud-init)

CSCS and OpenCHAMI

- Tech sovereignty
- Reduce vendor dependency
- Be part of its development
- Easy to adopt (based on CSM)

The challenge

The chasm

- ALPS has many different communities of users (CSCS, PSI, MCH)
- Tackle each community at a different stage
- Feature match between CSM and OpenCHAMI
- Adapt our workloads to OpenCHAMI
- Integrate vServices to OpenCHAMI

CSM vs OpenCHAMI

	CSM	OpenCHAMI
Cloud interface	Yes	Yes
Relies on Kubernetes	Yes	No
Microservice arch	Yes	Yes
Redfish integration	Yes	*
Hardware inventory	Yes	*
Power management	Yes	*
Image builder tool	Yes	Yes
Image build pipeline	Yes	No
Image inventory	Yes	No
Cluster defined as data	Yes	No
Node console	Yes	**
WLM provisioning	Yes	No
Network manager	Yes	No
Node grouping	Yes	Yes
Node runtime config	Yes (ansible)	Yes (cloud-init)
Boot management	Yes	Yes
Software attestation	Yes	**
IAM	Yes	**
Monitoring	Yes	**
Storage management	No	No

- *: Under review
- **: Under development

The plan

What can we do today? 2026 plan

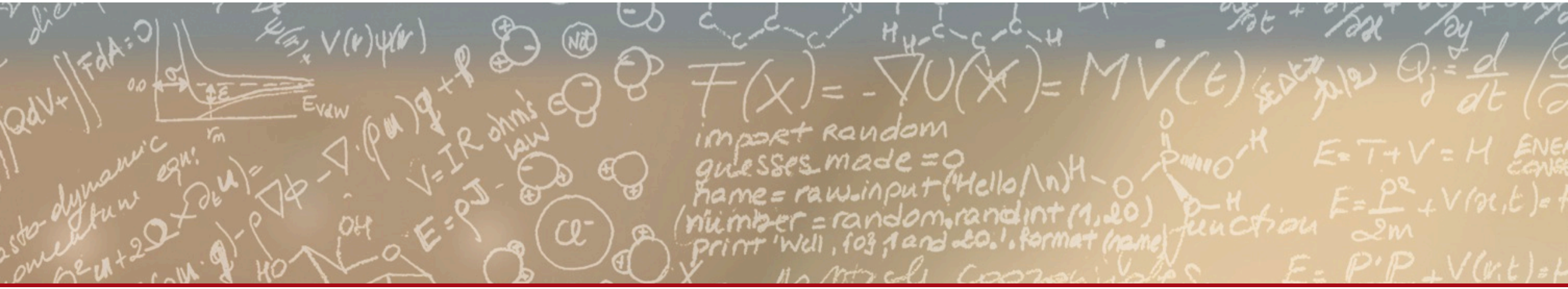
- Smooth transition for our tenants (CSCS clusters first then the rest)
- Having both OpenCHAMI and CSM deployed together
- Migrate compute nodes from CSM to OpenCHAMI
- Avoid shutting down the whole system
- OpenCHAMI deployment automation
- Integration tests
- CSCS-OpenCHAMI-vx.y.z
- Prepare test clusters for other working structures
- Coordinate with other working structures for:
  - Networking
  - Monitoring
  - Security
  - Clean hardware inventory from CSM and populate in OpenCHAMI

The process

- Communication across teams is highly important
- Make people aware of what is coming
- Changes in their processes (OpenCHAMI lack of features vs CSM)
- Use time as a reference

What is missing?

- Fleet manager
  - interface to manage multiple OpenCHAMI instances
- Image inventory
  - what images do I have available?
  - how was the image created?
  - where is the image stored?
  - who created the image?
  - when was the image created?
- Redfish integration
  - how can new hardware be integrated?
  - process to build Redfish clients compatible with OpenCHAMI
  - how could a vendor publish their Redfish clients
- Image build pipeline
  - normalize the way we create images and push images
- Cluster/Node configuration automation
  - data defined cluster
  - template that describes how my cluster should looks like
  - configuration inventory
  - rollback



Thank you for your attention.

https://www.cscs.ch
https://www.cscs.ch/computers/alps
https://openchami.org
https://github.com/openCHAMI
https://hpsf.io/projects/openchami
https://github.com/Cray-HPE