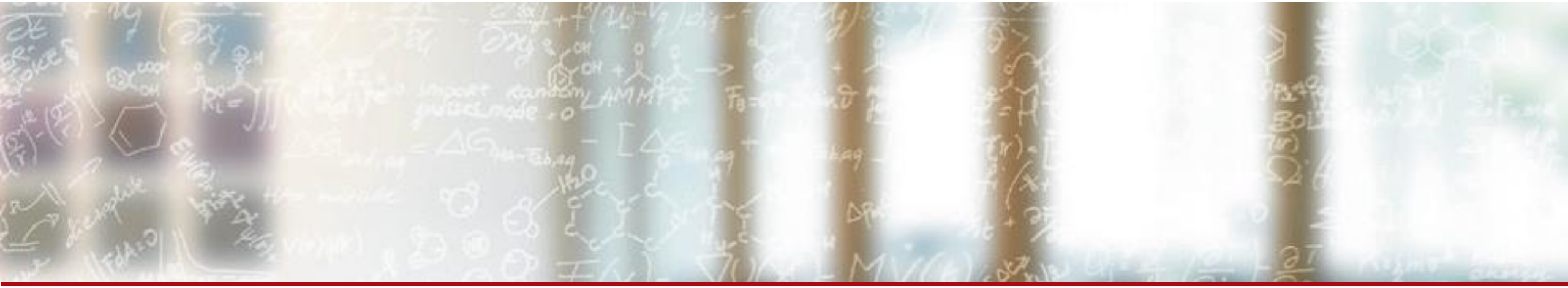
CSCS
Centro Svizzero di Calcolo Scientifico
Swiss National Supercomputing Centre

ETH zürich



Hyper-converged cloud infrastructure at CSCS

Dino Conciatore – System Engineer
Elia Oggian – System Engineer

Monday, May 5th, 2025

CUG 2025

Background

- Operating a **multi-tenant Kubernetes** cluster can be very complex
  - Based on past experience with *fulen*, a multi-tenant cluster with multiple VLANs and users with varying requirements and workflows
- Improvements were necessary to ensure a **greater user experience**
- **Simplify the deployment** of CSCS internal services
- **Security and Isolation**:
  - network isolation should be implemented per cluster or purpose

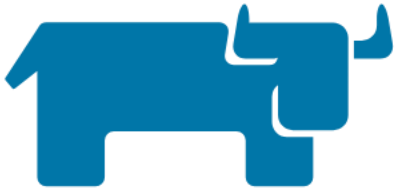
Introducing Key Components

Introduction to SUSE Virtualization (Harvester)

Harvester is an open-source hyperconverged infrastructure solution.

- **Kubernetes** as orchestration engine
- Based on **KVM** and **KubeVirt**, which enable virtualization and orchestration of VMs
- **Storage** management through Longhorn to provide a persistent storage
- **Networking** based on **Multus** enables isolation on multiple VLANs

Introduction to SUSE Rancher



Rancher is an open-source Kubernetes clusters manager.

- Centrally manage **multiple Kubernetes clusters**
- Simplified **provisioning and scaling** of clusters, node management, and upgrades.
- Strong **security features**, including role-based access control (RBAC), network policies and integration with identity providers to enhance security and compliance.
- Rancher has its own Kubernetes distribution called RKE2

Introduction to ArgoCD

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

- **Application deployments** are declared 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** for visualizing and managing application deployments status. Additionally, it offers a command-line interface (CLI) for scripting and automation.

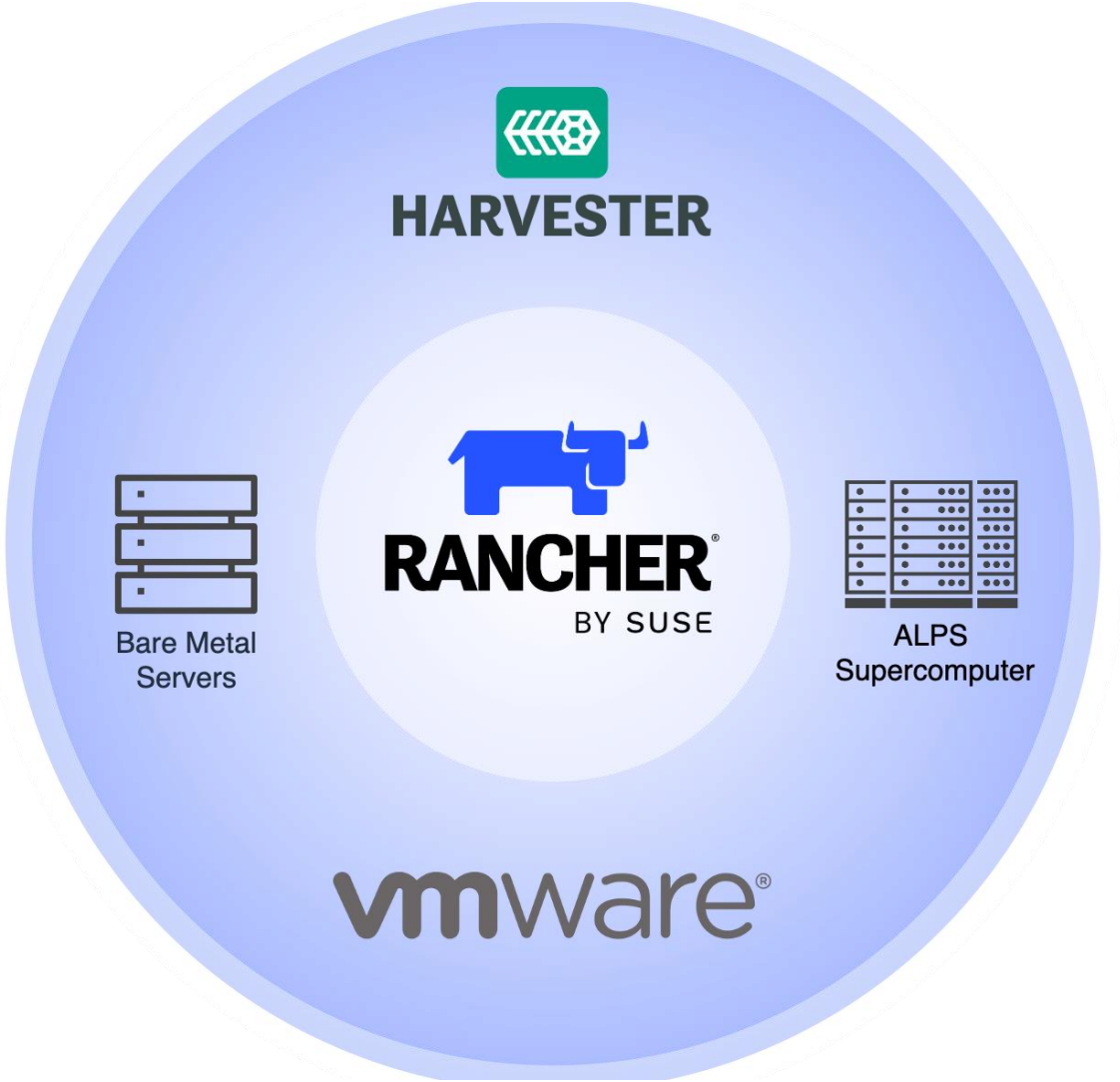
CSCS
Centro Svizzero di Calcolo Scientifico
Swiss National Supercomputing Centre

ETH zürich

Hyperconverged Cloud Infrastructure at CSCS

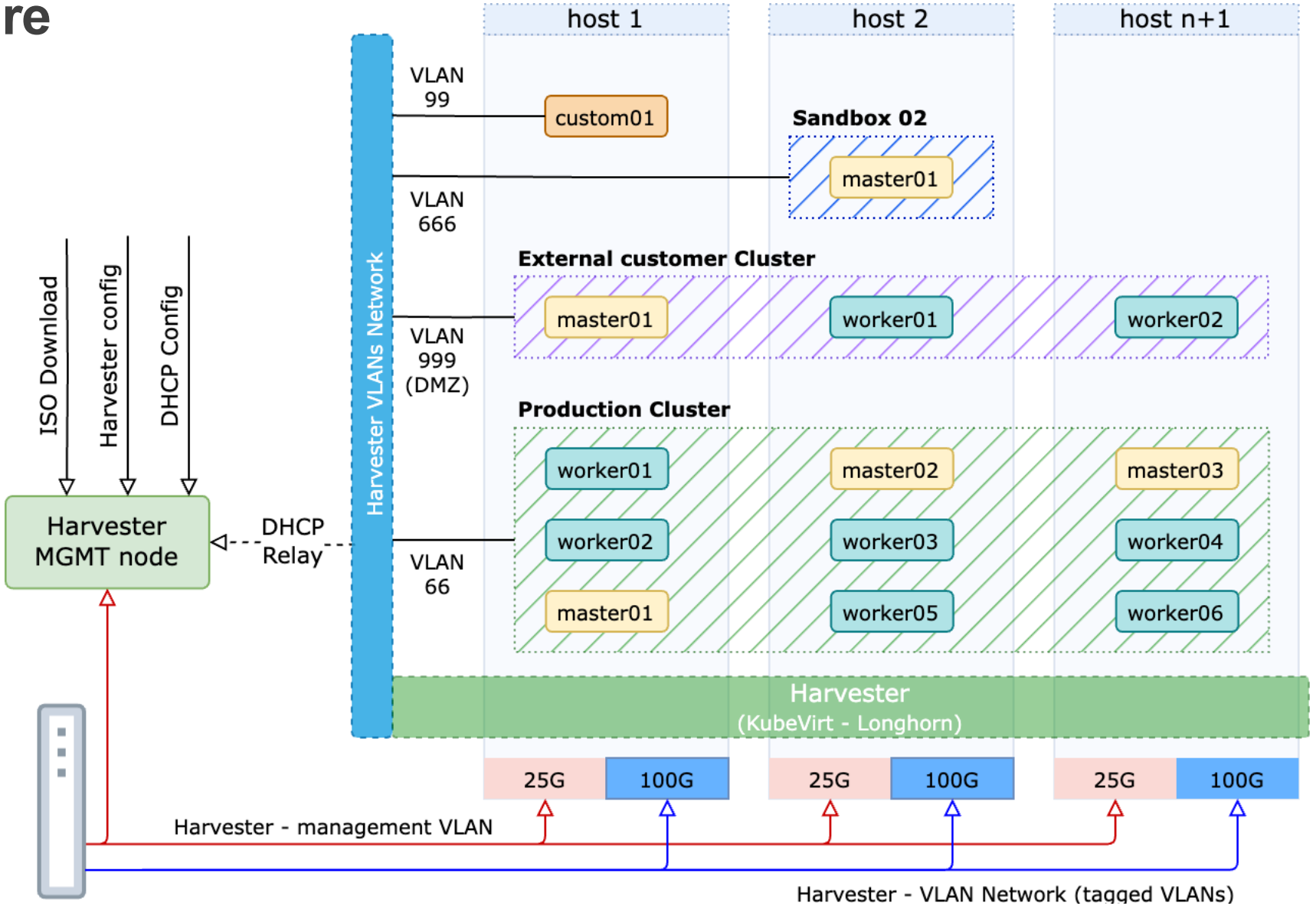
Where we run kubernetes

- 3 nodes
  - AMD EPYC 24 cores
  - 192G RAM
  - 2x 10G NIC (LACP)
- ~ 50 Downstream RKE2 Clusters
  - SUSE Virtualization
  - VMWare
  - Bare Metal
  - Alps (HPC)



Harvester architecture

- 16 Hosts
- Networks
  - Management (25G)
  - VLANs (100G)
- Local 7.68 TB NVMe
- 512G/768G RAM
- AMD EPYC 64 Cores
- Management node:
  - DHCP Relay for VMs
  - iPXE Boot
  - ISO Repository
  - Hosts config

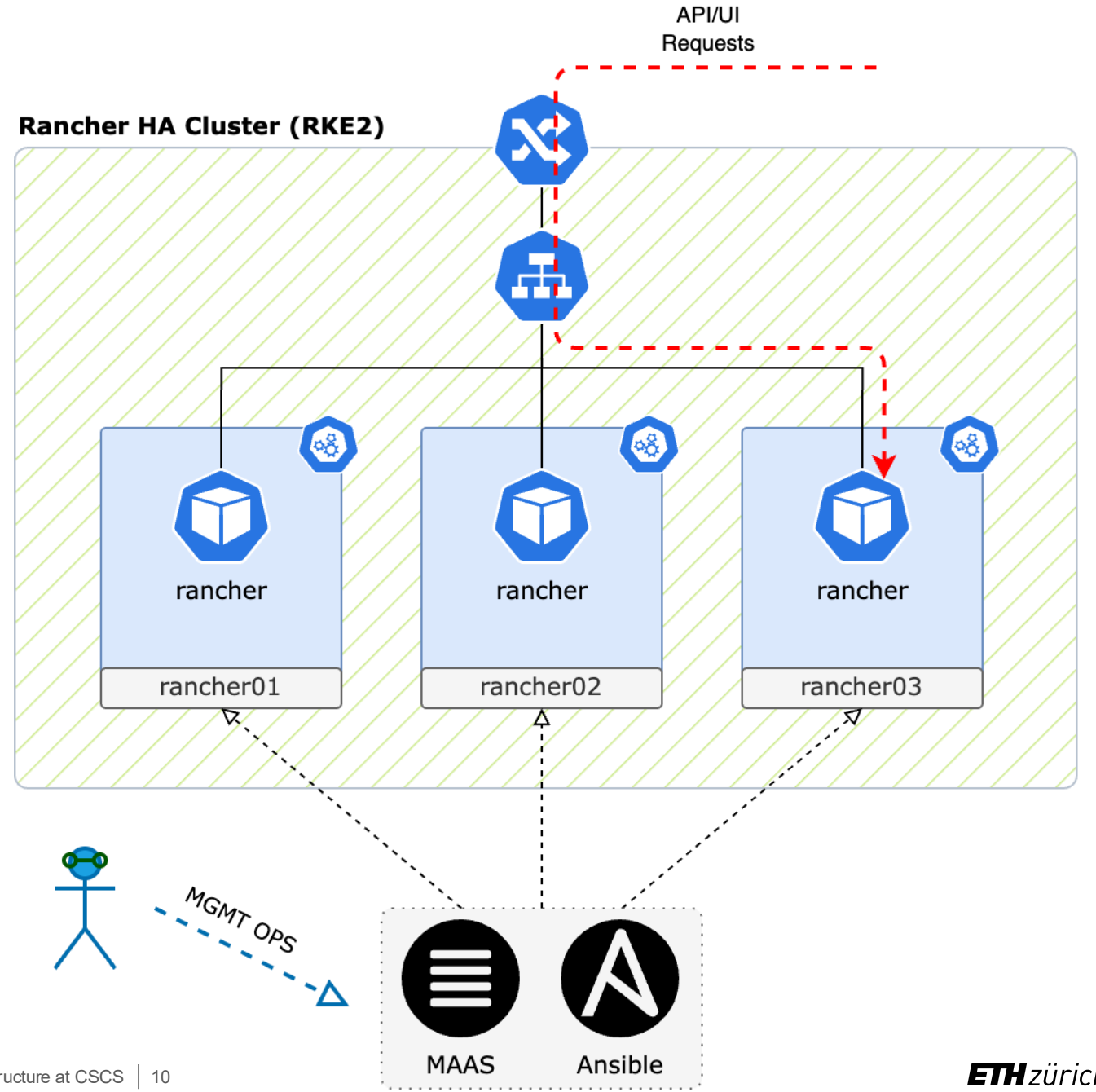


Rancher cluster architecture

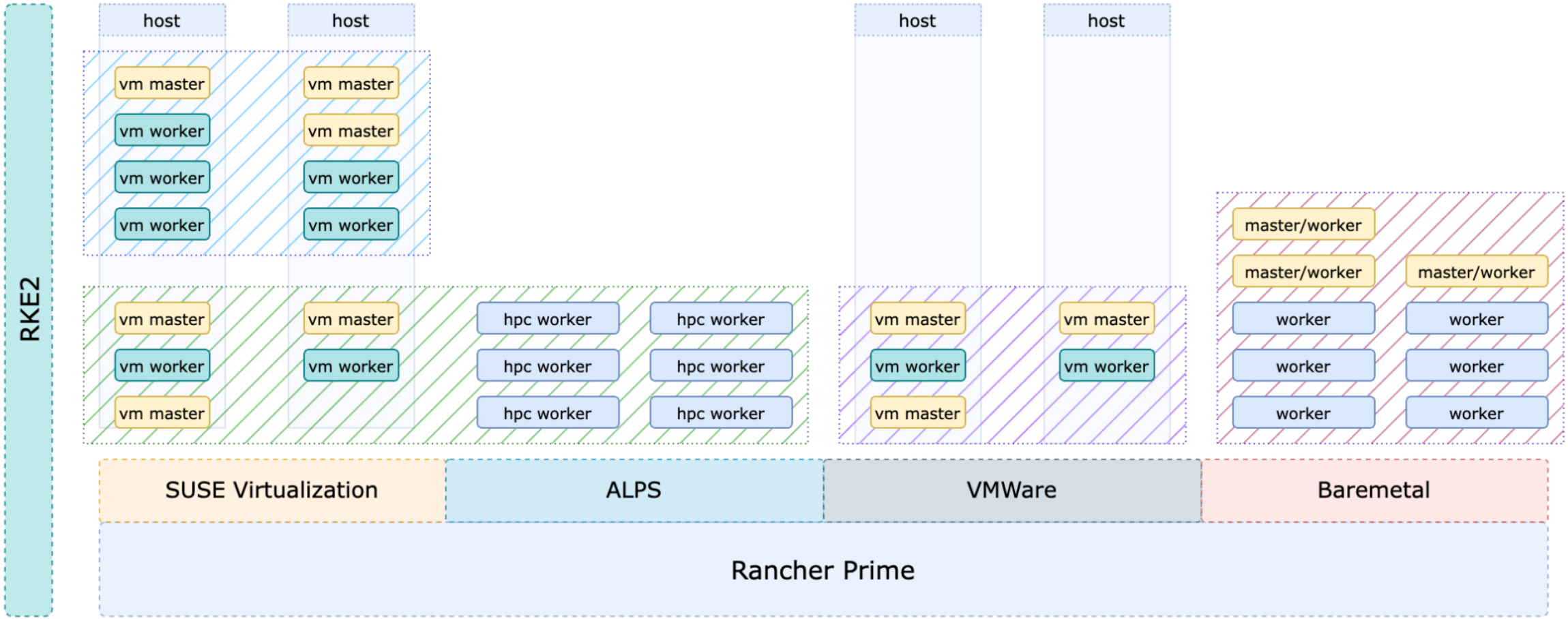
- RKE2 Cluster
- 3 nodes both master and worker
- Each node is expected to have a Rancher pod
- kube-vip enables external access to the UI/API

Node management:

- Node lifecycle through MAAS
- RKE2 installed and managed with Ansible
- Rancher deployment managed with Helm



Tenants overview

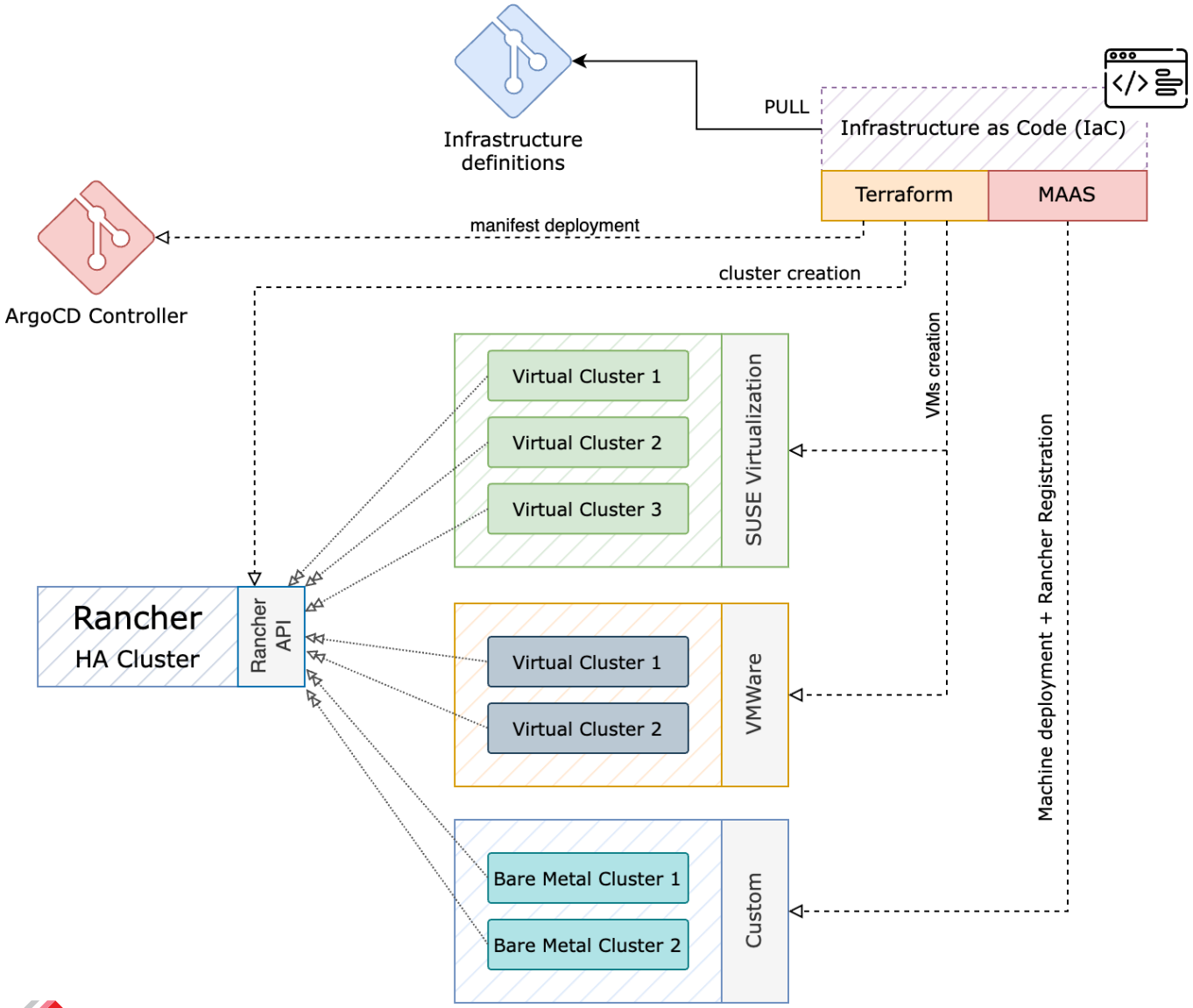


CSCS
Centro Svizzero di Calcolo Scientifico
Swiss National Supercomputing Centre

ETH zürich

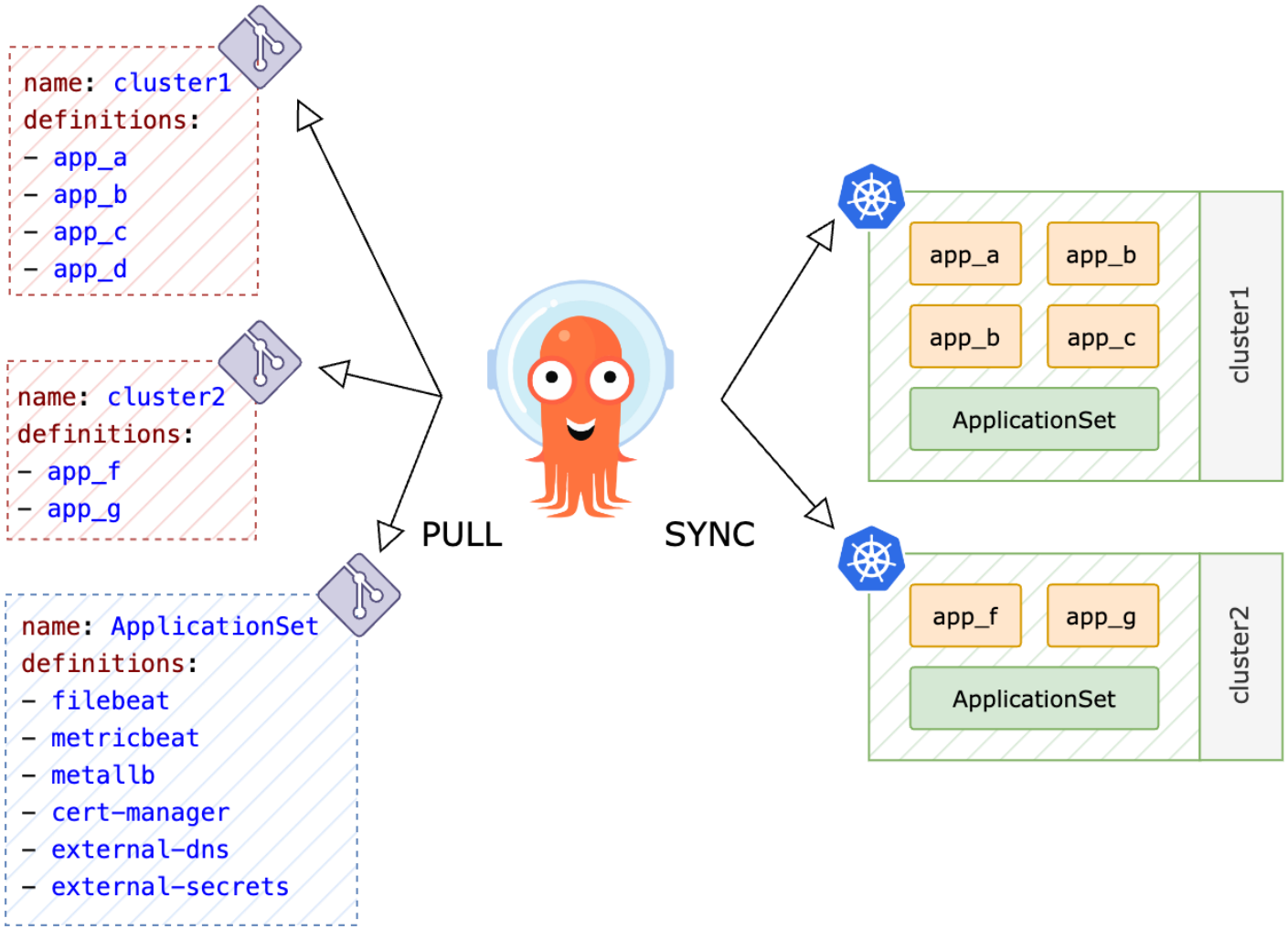
Infrastructure as a code and GitOps

IaC cluster provisioning



- Declarative approach
- Define cluster
  - Node specs
  - Network isolation
- Terraform
  - Community providers
  - CSCS cluster modules
  - Bootstrap RKE2 clusters

GitOps with ArgoCD

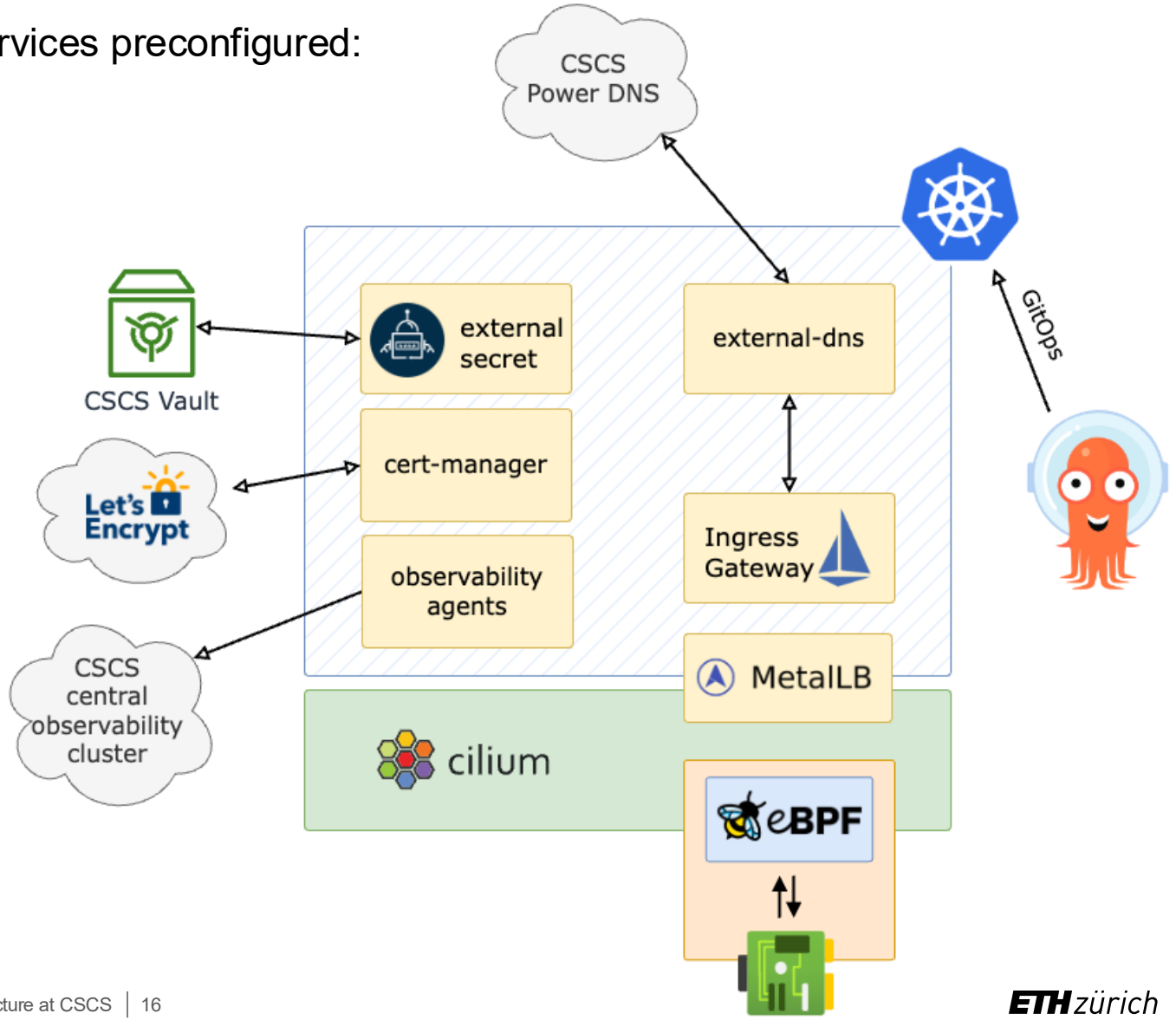


- Controller (git repo)
  - Clusters
  - Repositories
  - Projects
  - Applications
- ApplicationSets
  - CSI
  - Ingress Controller
  - Certificate Management
  - Observability
- App of Apps
  - Declarative deployments
  - 750 Applications across 50 clusters

Kubernetes clusters common features

All clusters deployed have the following common services preconfigured:

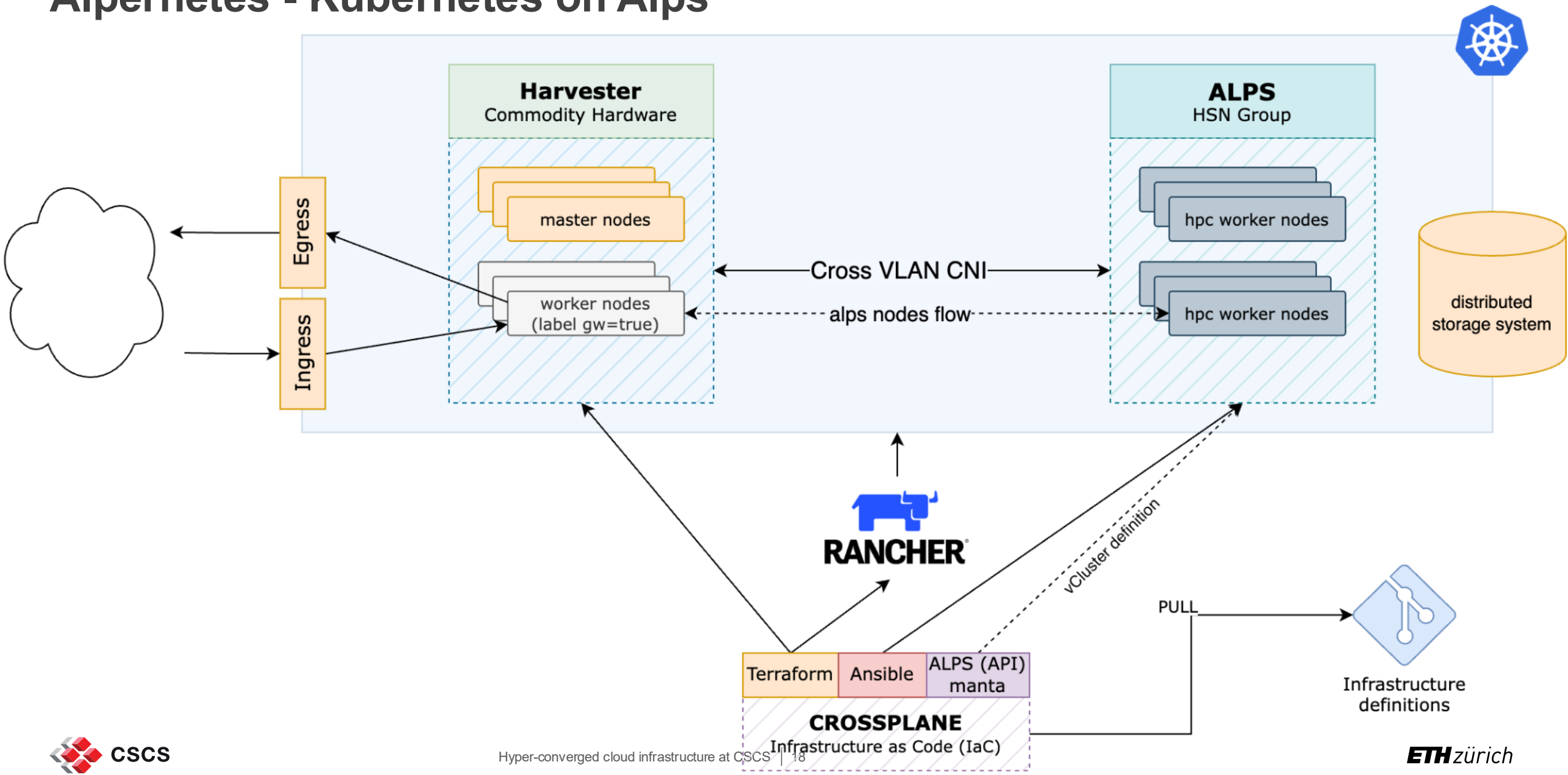
- CNI: Cilium
  - Service mesh (eBPF)
  - Hubble (observability UI)
- CSI:
  - cephfs
  - ceph rbd
- Ingress controller
  - Istio (no sidecars)
    - Started migration to new API Gateway
  - Nginx
- MetalLB
  - Gratuitous ARP
- Automated DNS and Certs
- External Secret
  - For all Vault secrets
- Observability agents
  - Filebeat and Metricbeat



Crossplane

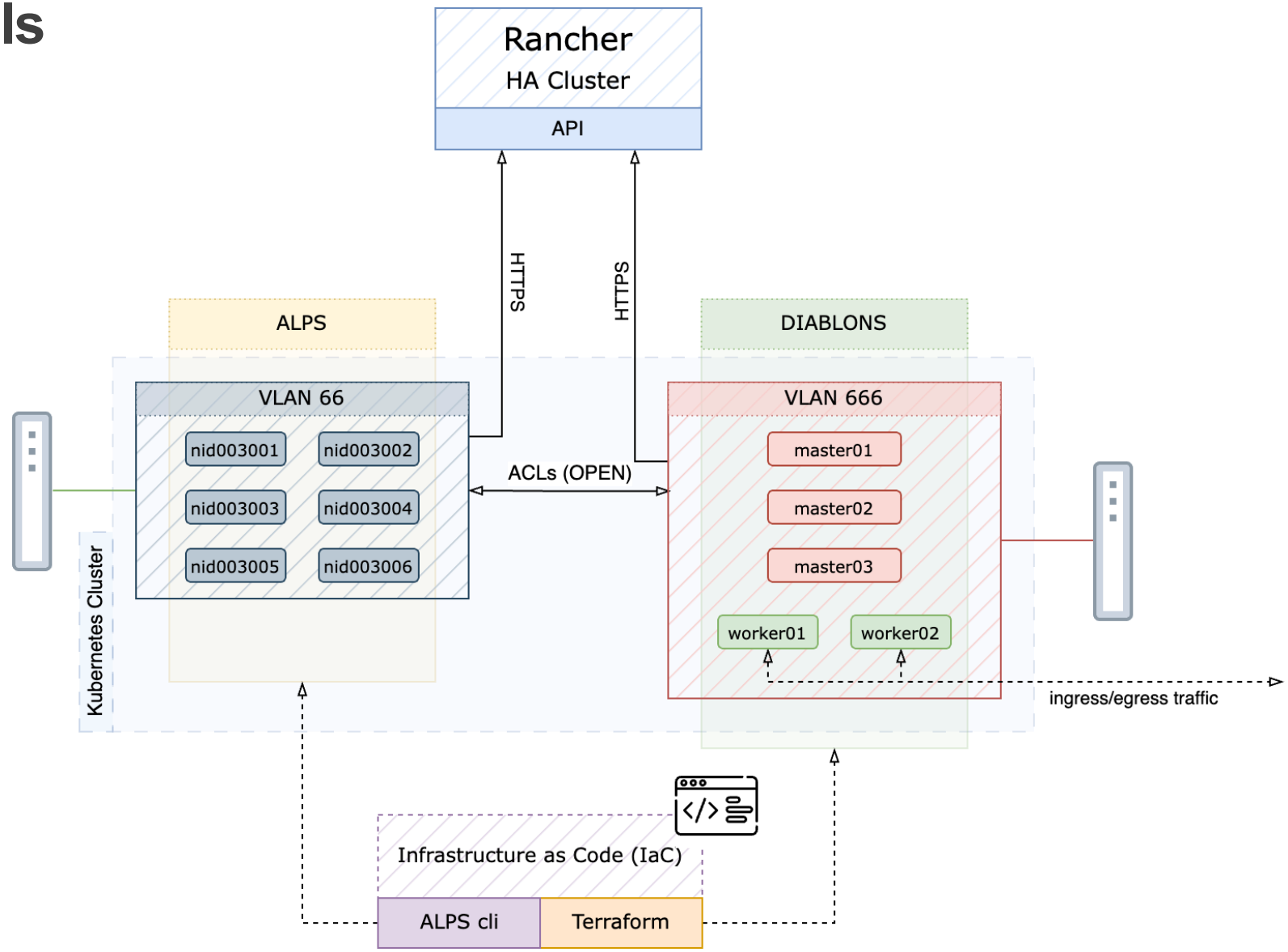
- **Open-Source Control Plane**
  - Manage cloud resources and services
- **Kubernetes Native**
  - Extends Kubernetes with cloud-native capabilities
- **Infrastructure as Code**
  - Define and manage infrastructure using Kubernetes API and CRDs
- **Multi-Cloud Support**
  - Compatible with AWS, Azure, and more
- **Declarative API**
  - Use Kubernetes manifests to describe infrastructure
- **Composable Infrastructure**
  - Modular and reusable infrastructure components
- **Secure and Consistent**
  - Enforces policies and ensures consistent configurations
- **Scalable and Extensible**
  - Easily scales and integrates with other tools

Alpernetes - Kubernetes on Alps

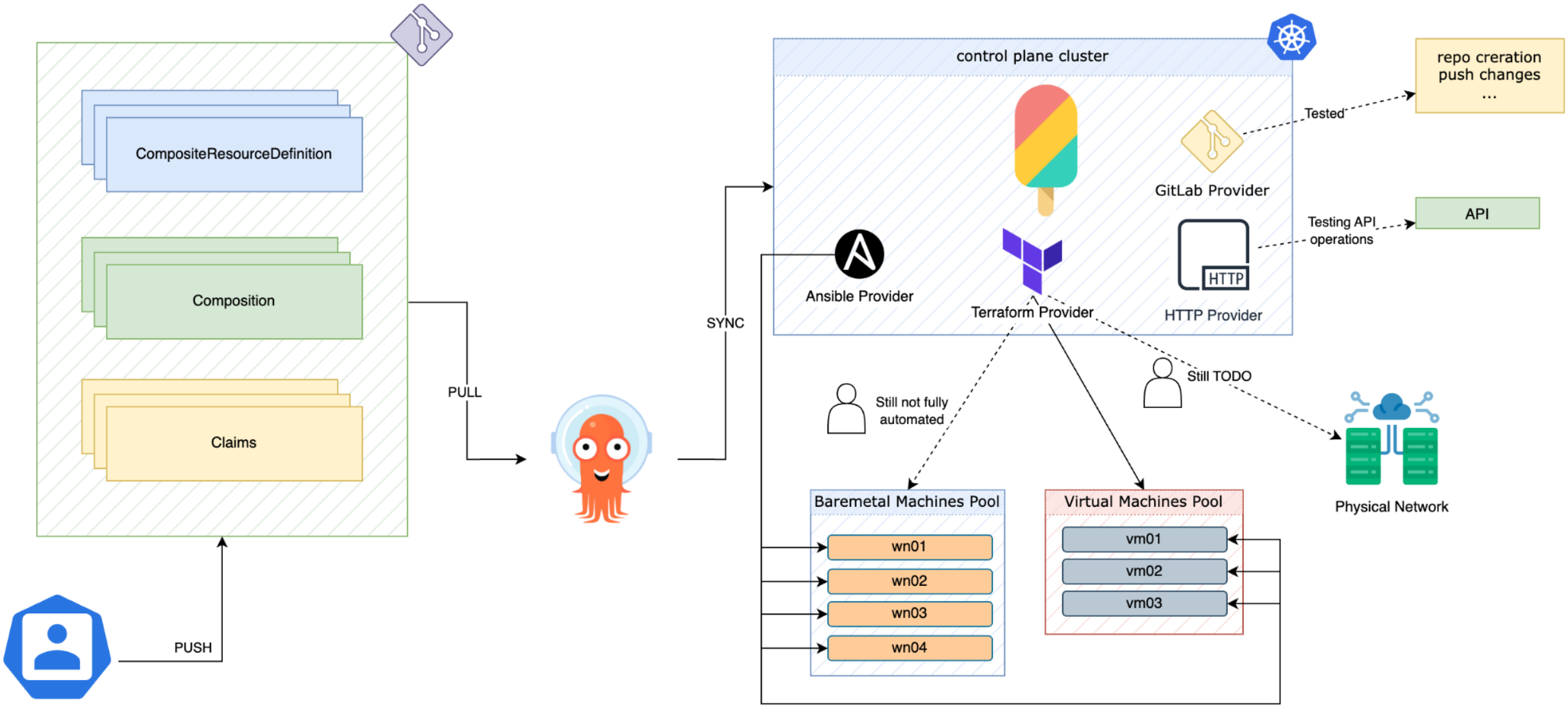


Kubernetes on alps details

- Harvester (DIABLONS)
  - Master nodes
  - Specific workload workers
    - Ingress Controller
    - Egress Controller
    - Low performance SVC
    - Etc…
- ALPS (HPC)
  - Stateless workernodes



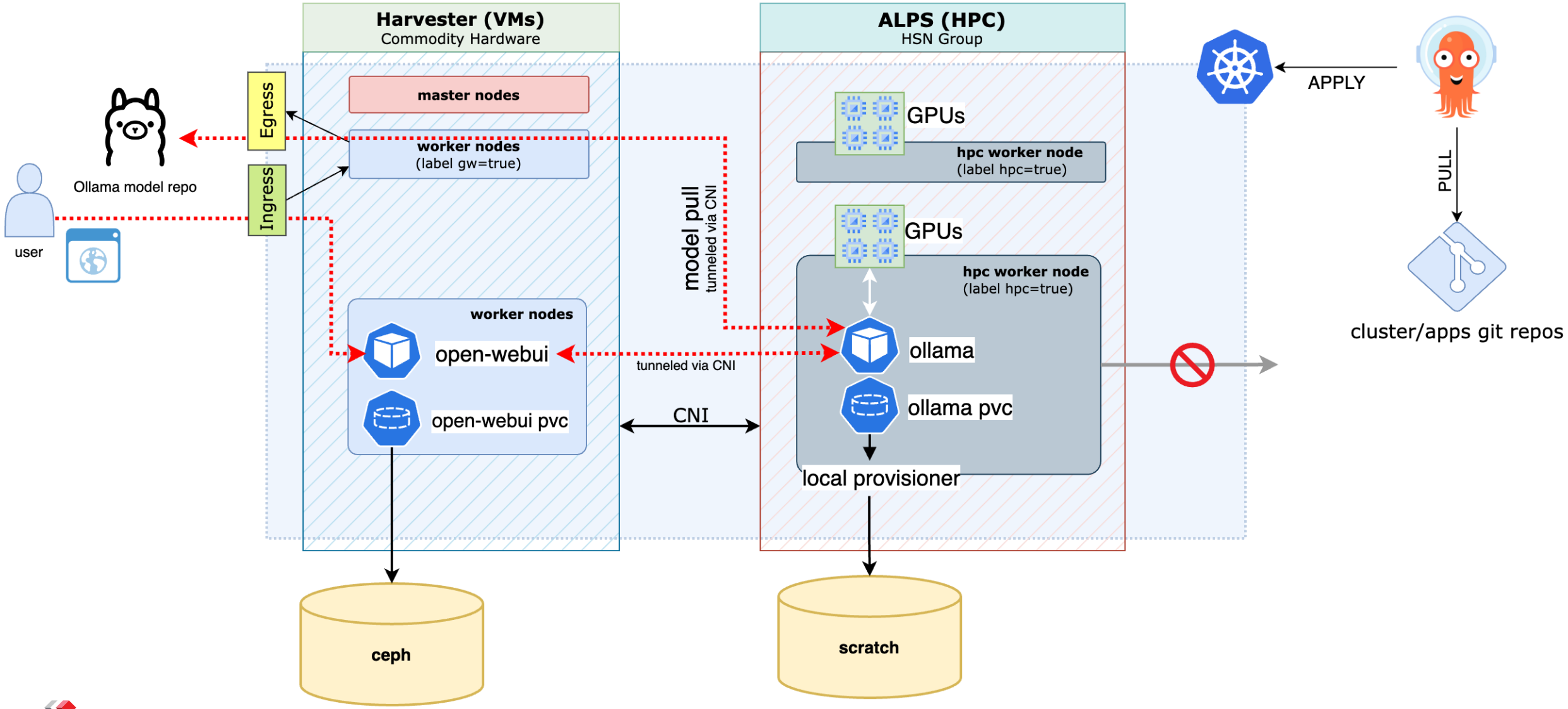
Alpernetes - Kubernetes on Alps



Crossplane claim example

```yaml
---
apiVersion: xplank.io/v1alpha1
kind: Alpernetes
metadata:
  name: demo-hpc
spec:
  clustername: demo-hpc
  kubernetes_version: v1.30.11+rke2r1
  local_auth_endpoint_fqdn: demo-hpc.tds.domain.ch
  network_vlan: "999"
  vms_map: |
    vms:
      - prefix: "worker"
        vm_count: 6
        disk_size: 50
        cores: 4
        memory: 8
      - prefix: "master"
        vm_count: 3
        disk_size: 50
        cores: 4
        memory: 8
  #Inventory
  hpc_workers: |
    [hpc_worker:vars]
    ansible_ssh_private_key_file=ansible_ed25519
    [hpc_worker]
    nid000001
    nid000002
    nid000003
    nid000004
```

Alpernetes use case example



Operations

Infrastructure upgrade

- **Harvester**
  - Upgrade triggered automatically via the user interface.
    - RKE2 upgrade
    - Draining one host at a time
      - VMs are live migrated
    - OS Image replacement
- **Rancher**
  - Ansible managed nodes:
    - Drain Kubernetes nodes one by one.
    - Reinstall or upgrade the OS
  - Rancher
    - Upgrade done via Ansible (same playbook used for the installation)
      - `rke2_version: v1.26.9+rke2r1`
      - **NOTE:** Don't skip intermediate minor versions when upgrading, eg. : *v1.28.x → 1.29.x*
- **ArgoCD**
  - via Helm

Downstream clusters upgrade

- **Kubernetes**
  - Required RKE2 version can be changed using Terraform

    ```
    kubernetes_version = "v1.29.2+rke2r1"
    ```

    ```
    $ terraform apply
    ```

    - **NOTE:** Don't skip intermediate minor versions when upgrading, eg. : *v1.28.x → 1.29.x*
  - Rancher will handle draining nodes and upgrading
- **OS Upgrades**
  - Harvester nodes (VMs):
    - Required OS can be changed using Terraform
      - `image = "default/ubuntu-jammy”`
      - `$ terraform apply`
  - Ansible managed nodes:
    - Drain Kubernetes nodes one by one.
    - Reinstall or upgrade the OS based on your requirements

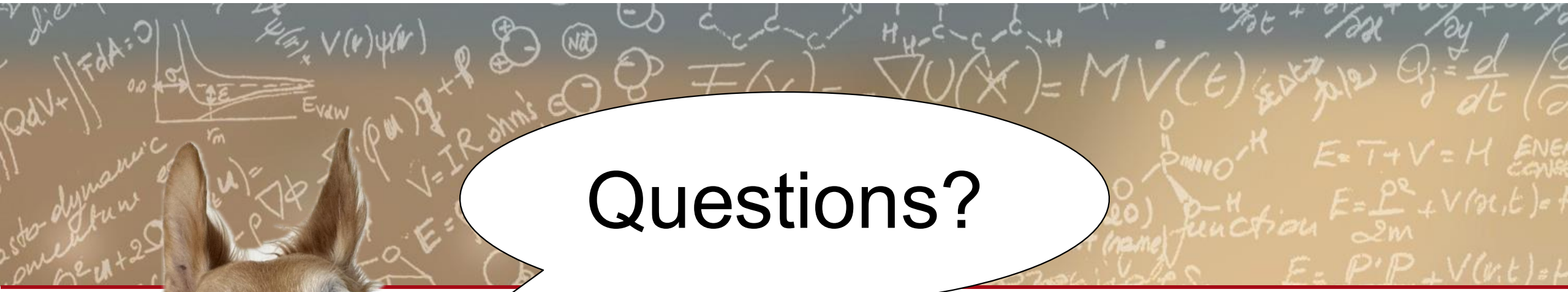
Use cases

Conclusion and Usecases

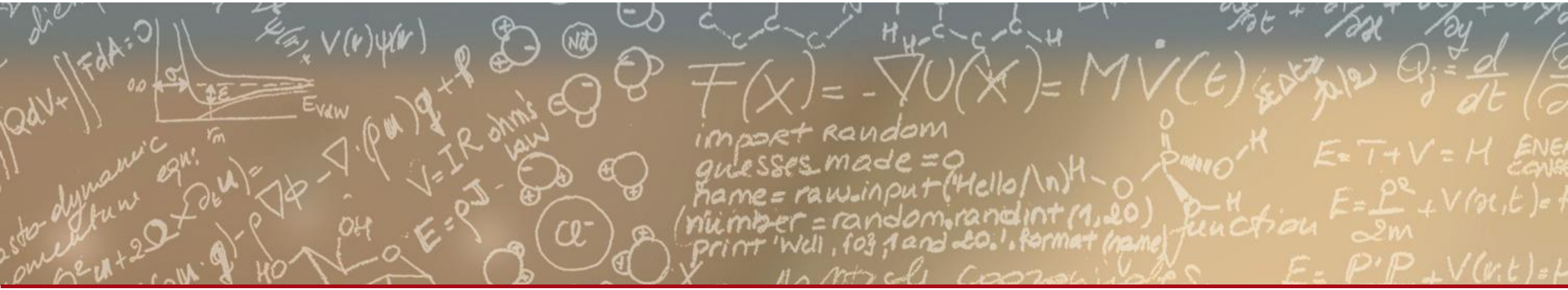
We are running a total of ~50 downstream clusters, with ~300 VMs and ~200 bare metal nodes

Here are some of the Kubernetes clusters:

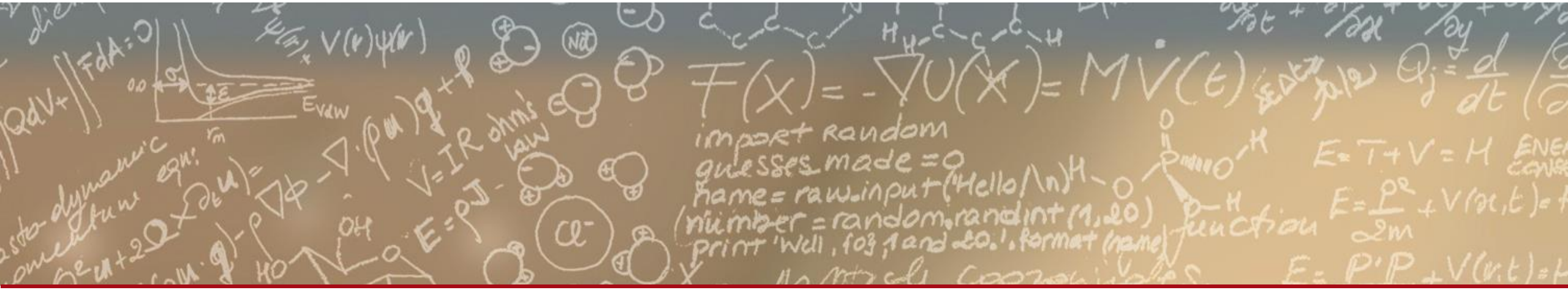
- dCache
  - 16 bare metal nodes
  - ~250 pools
  - 48TiB of PVCs via Ceph RBD (Total 12PB WLCG + CTA)
- WLCG Services
  - Frontier Squid
    - With multus and whereabouts (IPAM)
  - Nordugrid ARC CEs
  - site-bdii
- Observability cluster
  - 120 bare metal nodes
  - Local NVMe storage
    - Local Path for Elasticsearch data nodes
    - Longhorn for all other persistent services
- Central CSCS services
  - Vault
  - Jfrog
  - Gitlab-runners
- HPC Tests
  - Multicores cluster
  - GH200 cluster on RKE2



Questions?



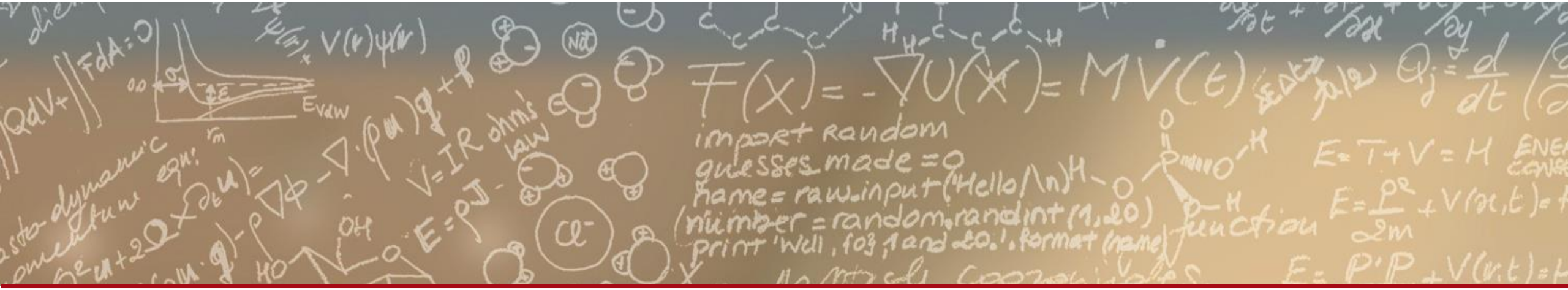
Thank you for your attention.



Backup slides…

Cilium tuning

```
rke2-cilium:
  bpf:
    masquerade: true
  egressGateway:
    enabled: true
  hubble:
    enabled: true
    relay:
      enabled: true
    ui:
      enabled: true
  ingressController:
    enabled: true
  k8sServiceHost: 127.0.0.1
  k8sServicePort: 6443
  kubeProxyReplacement: strict
```



Observability backup slides...

Observability

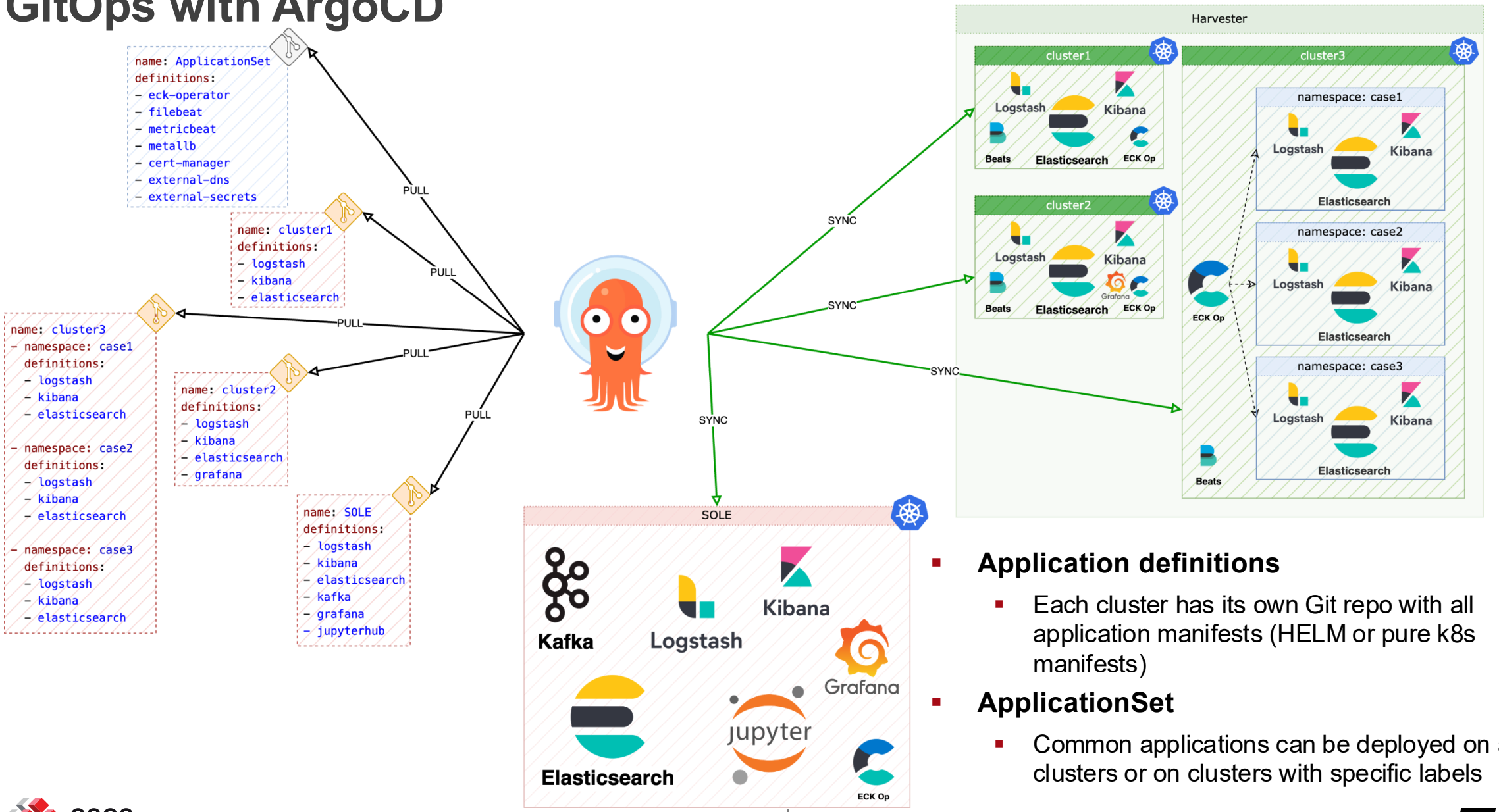
All clusters come with automatically deployed logs and metrics collectors (ArgoCD ApplicationSet), which send everything to our central Observability Infrastructure.

Currently we are using Elastic Beats

- Autodiscovery
  - Triggered on **`New`**, **`Modified`** and **`Deleted`** pod
- Hint based
  - Configure Beat modules via pod Annotations:

```
annotations:
        co.elastic.metrics/hosts: '"${data.host}:9404"'
        co.elastic.metrics/metrics_path: /metrics
        co.elastic.metrics/metricsets: collector
        co.elastic.metrics/module: prometheus
        co.elastic.metrics/processors.1.add_fields.fields.namespace: kafka
        co.elastic.metrics/processors.1.add_fields.target: data_stream
```

GitOps with ArgoCD



- **Application definitions**
  - Each cluster has its own Git repo with all application manifests (HELM or pure k8s manifests)
- **ApplicationSet**
  - Common applications can be deployed on all clusters or on clusters with specific labels

Backup slide: Elastic Stack flow