Ministero dell'Università e della Ricerca | Italiadomani PIANO NAZIONALE DI RIPRESA E RESILIENZA | PNC Piano nazionale per gli investimenti complementari al PNRR Ministero dell'Università e della Ricerca | DARE DIGITAL LIFELONG PREVENTION

High availability Kubernetes for security purposes

Francesco Sinisi (francesco.sinisi@cnaf.infn.it)
Ahmad Alkhansa (ahmad.alkhansa@cnaf.infn.it)
Alessandro Costantini (alessandro.costantini@cnaf.infn.it)
Diego Michelotto (diego.michelotto@cnaf.infn.it)
Barbara Martelli (barbara.martelli@cnaf.infn.it)

INFN

Outline

Ministero dell'Università e della Ricerca | Italiadomani PIANO NAZIONALE DI RIPRESA E RESILIENZA | PNC Piano nazionale per gli investimenti complementari al PNRR Ministero dell'Università e della Ricerca
INFN
DARE DIGITAL LIFELONG PREVENTION

Context

DARE project and connection with INFN

DARE (DigitAl lifelong pRevEntion) is a project part of the Italian National Recovery and Resilience Plan (NRRP). The initiative, taking advantage of new digital technologies, is aimed at creating and developing a community of knowledge, connected and distributed, which encourages the establishment of models and solutions for surveillance, prevention, health promotion and health safety.

The collaboration with INFN includes both the supply of **computational resources** (hardware) and **technical support** for their use and the **technologies** that can be deployed on them.

Considering the special nature of the data, i.e. **personal data** belonging to the medical field, we need a **secure infrastructure**, capable of dealing with any malicious attacks coming from the outside. Our platform was designed to meet these security needs.

Example use cases

IOR Solution

The aims are to estimate the risk of a **femur fracture** occurring when specific loads from different angles are applied and to carry out studies on the wear of knee prosthesis, by simulating joint kinematics and loading conditions over time.

The activity has been focused on investigating alternatives environment solutions in order to run workflows designed by researchers from the **Rizzoli Orthopedic Institute** (**IOR**).

Sant'Orsola Solution

The aim of the research project is to develop a workflow for the detection of **germline variants** on whole genome sequencing data.

The platform has been deployed on EPIC (Enhanced PrIvacy and Compliance) Cloud, where some sample genomic pipelines have been implemented and the needed security measures have been adopted to guarantee **GDPR compliance**.

Kubernetes HA deployment

Ministero dell'Università e della Ricerca | Italiadomani PIANO NAZIONALE DI RIPRESA E RESILIENZA | PNC Piano nazionale per gli investimenti complementari al PNRR *Ministero dell'Università e della Ricerca* | DARE DIGITAL LIFELONG PREVENTION

VM in Cloud with Openstack

Cluster nodes are made up of VMs. The advantages of using cloud resources are:

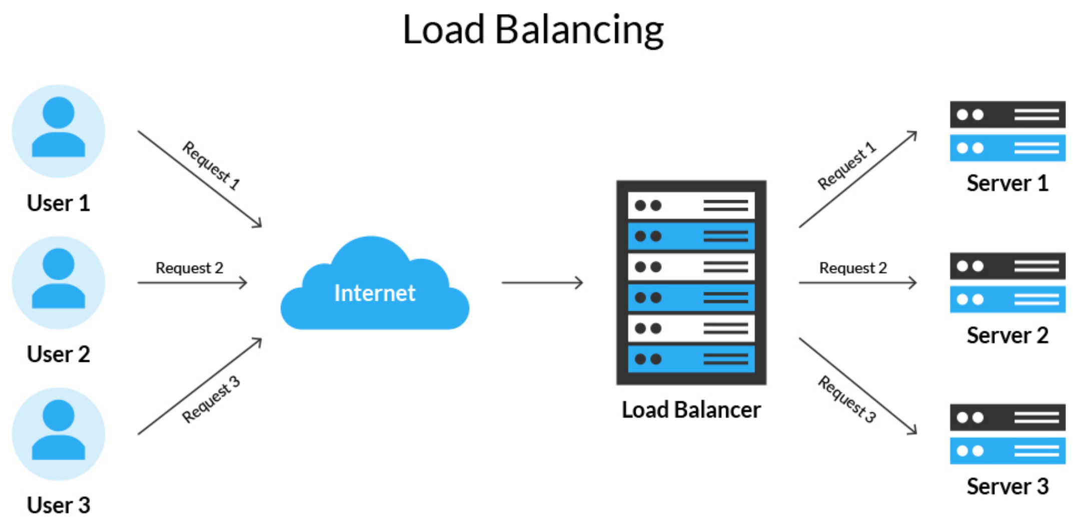
- easy and fast node creation (**horizontal scalability**)
- possibility of increasing the computational resources of a node belonging to the cluster (**vertical scalability**)

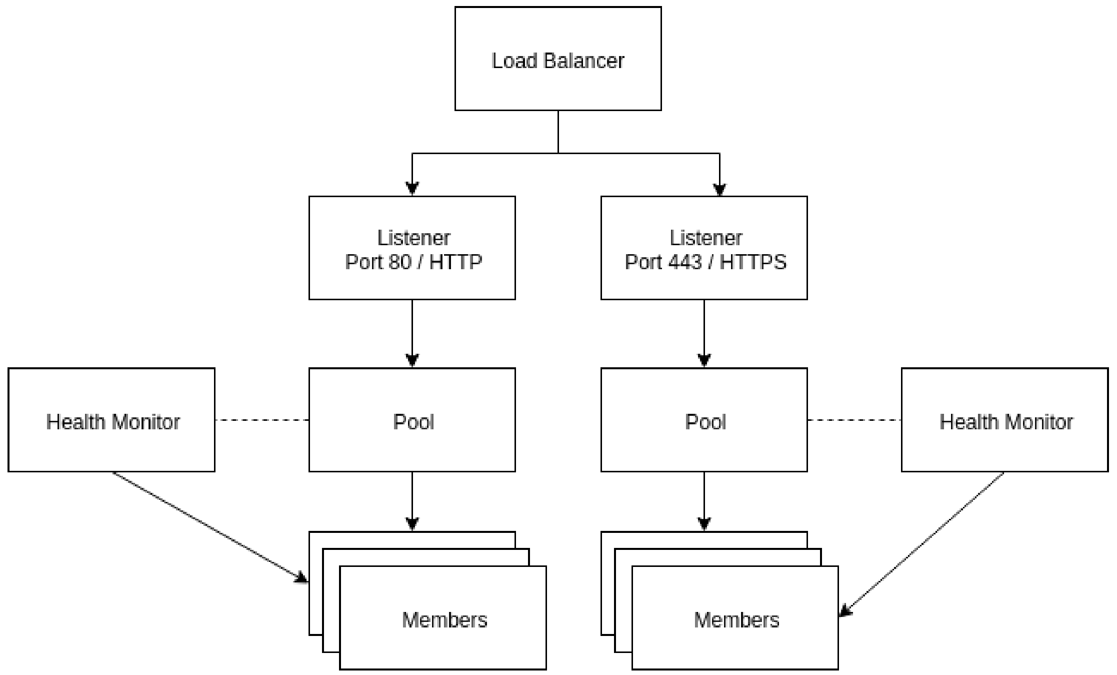
Tips for creating a good K8s cluster:

- Created masters and workers on different Hypervisors (**anti-affinity**)
- 3 masters on **SSD** (disk speed important to keep up with ETCD I/O)

Octavia LB – feature (1)

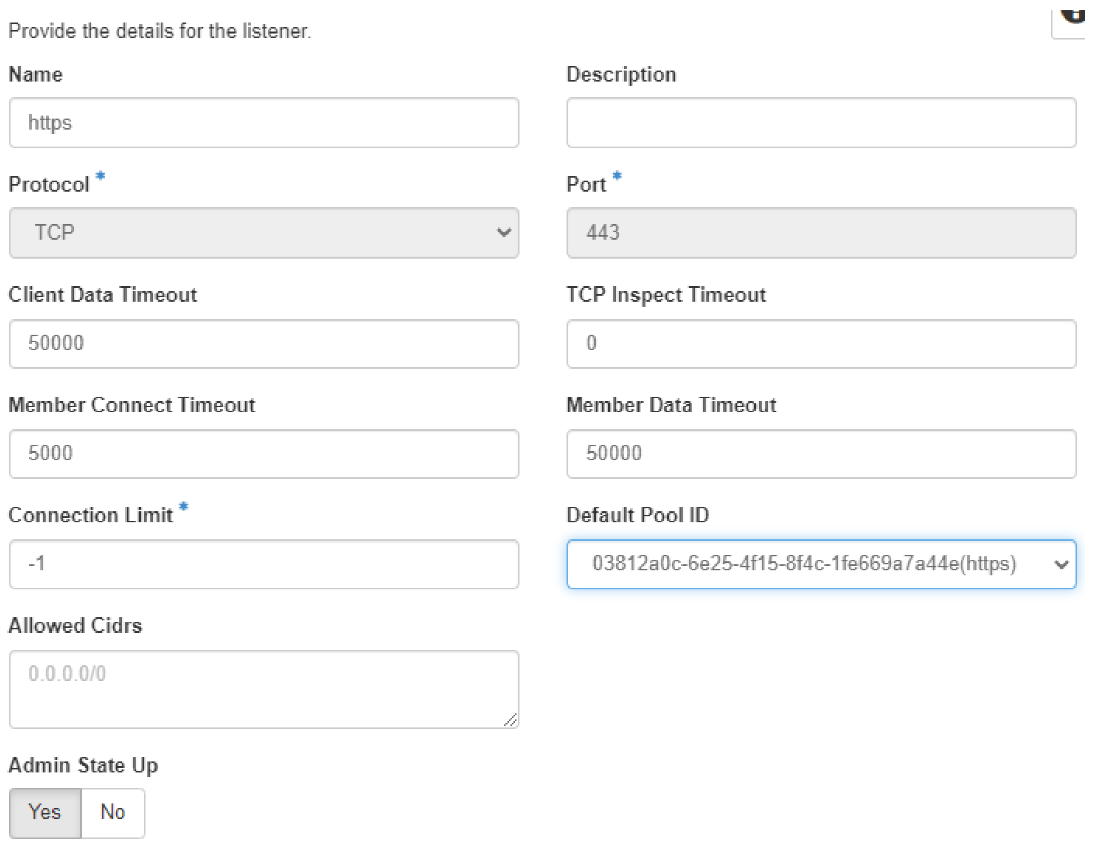
- Automatic creation of 2 VMs (active + backup)
- Health monitor integrated and pool composed of multiple members





Octavia LB – feature (2)

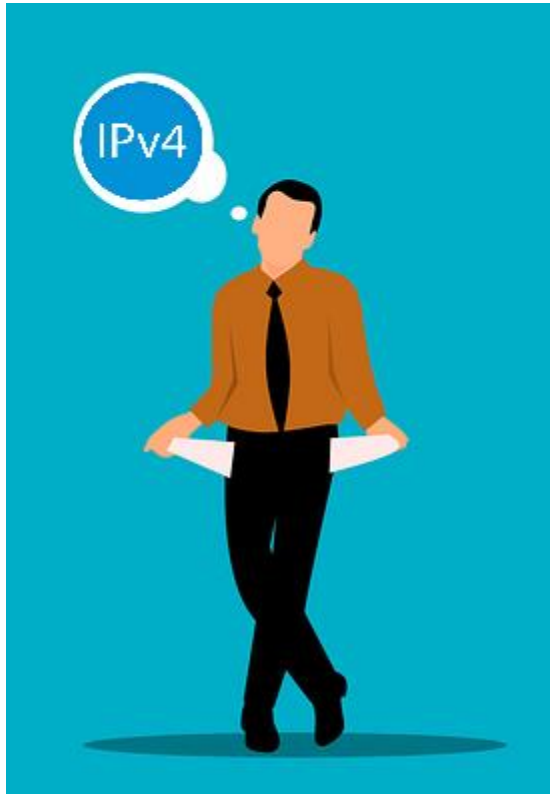
- Ability to easily configure connections (protocol, timeout, CIDR, etc.)



Provide the details for the listener.

Name	Description
https	
Protocol *	Port *
TCP	443
Client Data Timeout	TCP Inspect Timeout
50000	0
Member Connect Timeout	Member Data Timeout
5000	50000
Connection Limit *	Default Pool ID
-1	03812a0c-6e25-4f15-8f4c-1fe669a7a44e(https)
Allowed Cidrs	
0.0.0.0/0	
Admin State Up	
[✓ Yes] No	

- Lower consumption of FloatingIP

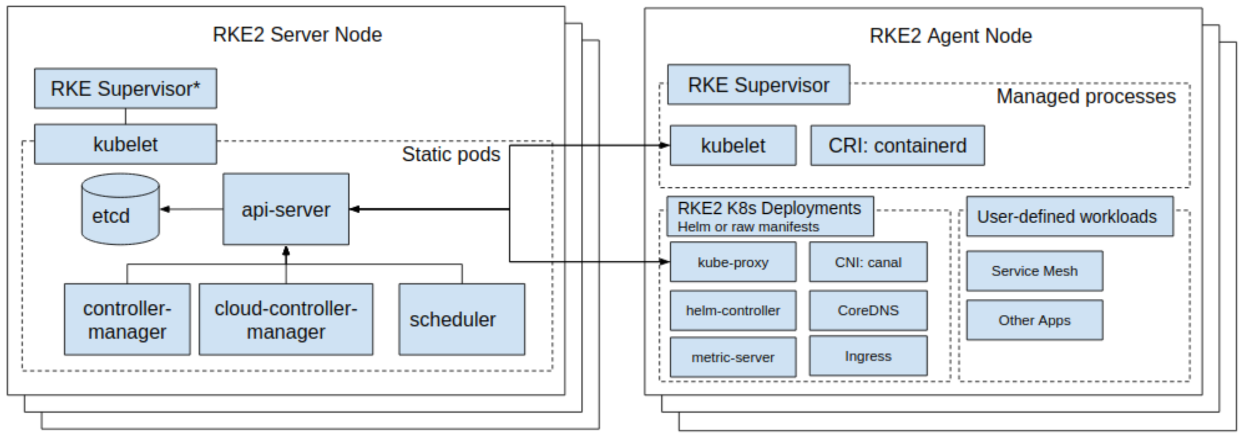


RKE2 – overview (1)

RKE2 is a tool to automate the creation of a K8s cluster. It can boast a large **community** behind, quick software updates, periodic scanning for **CVEs**, good documentations.

It installs software like CRI, CNI, helm, ingress Nginx, metric-server, kube-proxy, etc.

Launches a daemon on servers and workers that supervises the main components of the cluster (kubelet, etcd, controller, scheduler, helm), created as static Pods.



RKE2 – hardening (2)

One of the advantages of using RKE2 is the ability to easily create a hardened K8s cluster with a simple setup. It outlines the configurations and controls required to address Kubernetes benchmark controls from the **Center for Internet Security** (CIS).

RKE2 is designed to be "hardened by default" and pass most of Kubernetes CIS controls without modification. Other security checks are satisfied by using the CIS profile. There are, instead, a few notable exceptions to this, left to the discretion of the administrator, that require manual intervention to fully pass the CIS Benchmark.

The most important changes adopted, both automatic and manual, are:

- ensure **protect-kernel-defaults** is set (kubelet flag that will cause the kubelet to exit if the required kernel parameters are unset or are set to values that are different from the kubelet's defaults)
- ensure ETCD is configured properly (data directory be owned by the etcd user and group)
- limit **traffic** between namespaces (with some exceptions) with GlobalNetworkPolicy
- avoiding the creation of **default service accounts**

Puppet automation

RKE2 offers a bash script to run on the machine. We have extracted only the code useful for our purposes and encapsulated it in a puppet module. This serves to maintain the cluster state and re-create the cluster **as quickly as possible** in the rare case of a complete downtime.

The module has the following characteristics:

- Management of the daemon (server, agent) and various software based on the role
- Repository management based on k8s version
- Version lock management
- Package management based on version and role
- Management of configuration files based on environment parameters

Auxiliary software

Kyverno

Security implies limitations, which must be consciously circumvented by the system administrator. All this involves manual work, which can be avoided thanks to special automation.

In this regard, we use Kyverno. It is a policy engine that perpetually listens to Kubernetes APIs via a webhook and, when a new namespace is created, automatically creates and sets the following:

- Resource quota at the namespace level
    - Persistent (PVC) and ephemeral (Pod) storage
    - CPU and Memory requests
    - CPU and Memory limits
- Limit range at container level
    - Define the default CPU and Memory requests
    - Limitation on the maximum value of CPU and Memory
- Network policy
    - Allow traffic between workloads in the same namespace

Kyverno

Security implies limitations, which must be consciously circumvented by the system administrator. All this involves manual work, which can be avoided thanks to special automation.

In this regard, we use Kyverno. It is a policy engine that perpetually listens to Kubernetes APIs via a webhook and, when a new namespace is created, automatically creates and sets the following:

- Resource quota at the namespace level
  - Persistent (PVC) and ephemeral (Pod) storage
  - CPU and Memory requests
  - CPU and Memory limits
- Limit range at container level
  - Define the default CPU and Memory requests
  - Limitation on the maximum value of CPU and Memory
- Network policy
  - Allow traffic between workloads in the same namespace

```
- name: sync-resource-quota
  kind: ResourceQuota
  data:
    requests.cpu: 8
    requests.memory: 16Gi
    limits.cpu: 8
    limits.memory: 16Gi
    requests.storage: 10Gi
    requests.ephemeral-storage: 10Gi
```

Kyverno

Security implies limitations, which must be consciously circumvented by the system administrator. All this involves manual work, which can be avoided thanks to special automation.

In this regard, we use Kyverno. It is a policy engine that perpetually listens to Kubernetes APIs via a webhook and, when a new namespace is created, automatically creates and sets the following:

- Resource quota at the namespace level
  - Persistent (PVC) and ephemeral (Pod) storage
  - CPU and Memory requests
  - CPU and Memory limits
- Limit range at container level
  - Define the default CPU and Memory requests
  - Limitation on the maximum value of CPU and Memory
- Network policy
  - Allow traffic between workloads in the same namespace

```
- name: sync-limit-range
    kind: LimitRange
        - default:
            cpu: 500m
            memory: 4Gi
          defaultRequest:
            cpu: 100m
            memory: 2Gi
          max:
            cpu: "4"
            memory: 12Gi
          type: Container
```

Kyverno

Security implies limitations, which must be consciously circumvented by the system administrator. All this involves manual work, which can be avoided thanks to special automation.

In this regard, we use Kyverno. It is a policy engine that perpetually listens to Kubernetes APIs via a webhook and, when a new namespace is created, automatically creates and sets the following:

- Resource quota at the namespace level
  - Persistent (PVC) and ephemeral (Pod) storage
  - CPU and Memory requests
  - CPU and Memory limits
- Limit range at container level
  - Define the default CPU and Memory requests
  - Limitation on the maximum value of CPU and Memory
- Network policy
  - Allow traffic between workloads in the same namespace

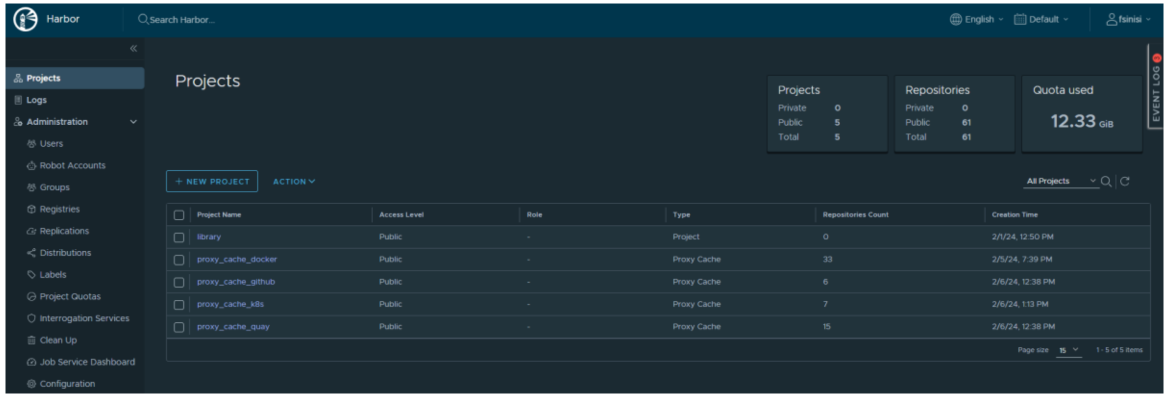
```
- name: sync-network-policy
  generate:
    kind: NetworkPolicy
    name: net-pol
    clone:
      namespace: default
      name: default-network-policy
```

Harbor registry

Created a Harbor instance to **cache** downloaded images on cluster nodes. In this way, in the rare case of a downtime of the entire cluster, when re-creating the cluster, we avoid having to make several requests to the official image registries, which usually impose a cap on downloads per unit of time.

To overcome this inconvenience, projects have been created to deposit images for the most famous repositories (DockerHUB, Quay.io, GitHub Container Registry and K8S Registry).

How does Registry Harbor work? RKE2 is configured to pull Docker images from the repositories listed above via Harbor. If it doesn't exist, the image is downloaded from the official repositories and cached on Harbor, then **proxyed** to the cluster.



Projects

Projects	
Private	0
Public	5
Total	5

Repositories	
Private	0
Public	61
Total	61

Quota used 12.33 GiB

Project Name	Access Level	Role	Type	Repositories Count	Creation Time
library	Public	-	Project	0	2/1/24, 12:50 PM
proxy_cache_docker	Public	-	Proxy Cache	33	2/5/24, 7:39 PM
proxy_cache_github	Public	-	Proxy Cache	6	2/6/24, 12:38 PM
proxy_cache_k8s	Public	-	Proxy Cache	7	2/6/24, 1:13 PM
proxy_cache_quay	Public	-	Proxy Cache	15	2/6/24, 12:38 PM

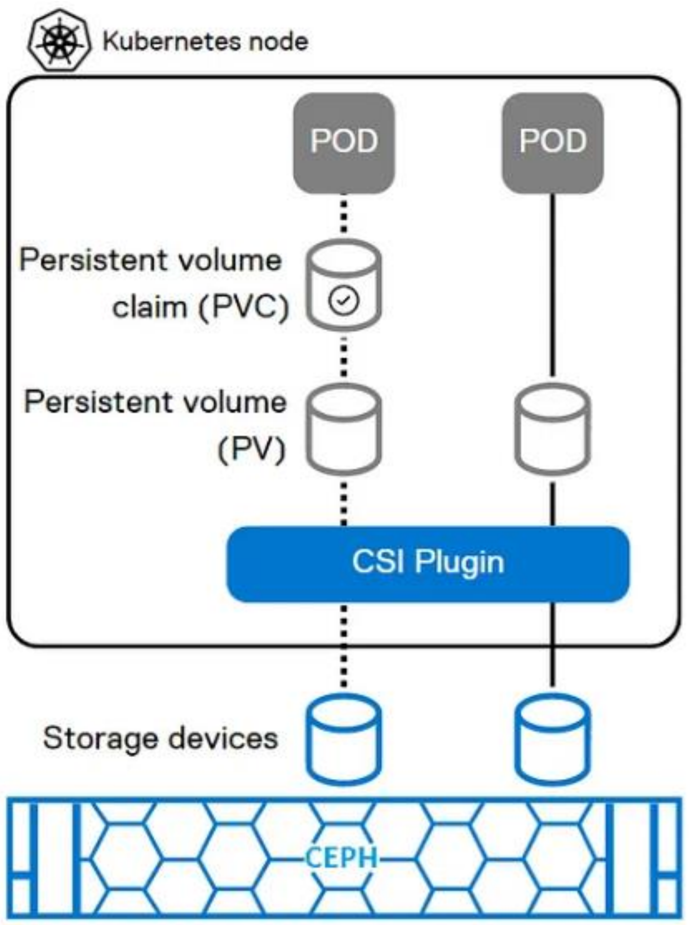
Page size 15 1 - 5 of 5 items

CSI CephFS

One of the requirements to obtain the CIS certificate is to avoid the use of **HostPath**, i.e. not to mount files contained in the host nodes in the Pod. For this reason, we have decoupled storage from the applications present on the cluster. As a storage solution we chose CephFS.

To allow communication between the 2 technologies it is necessary to configure:

- CephFS side
  - creation of a **volume**, which corresponds to the ceph file system
  - creation of a **subvolumegroup** for each cluster with related authorization and quota
- Kubernetes side
  - installation of the **CSI CephFS plugin**
  - Configuring quotas for namespaces (Kyverno)
  - Dynamic provisioner that receives requests from Kubernetes server APIs to manage CephFS subvolumes



Argo CD

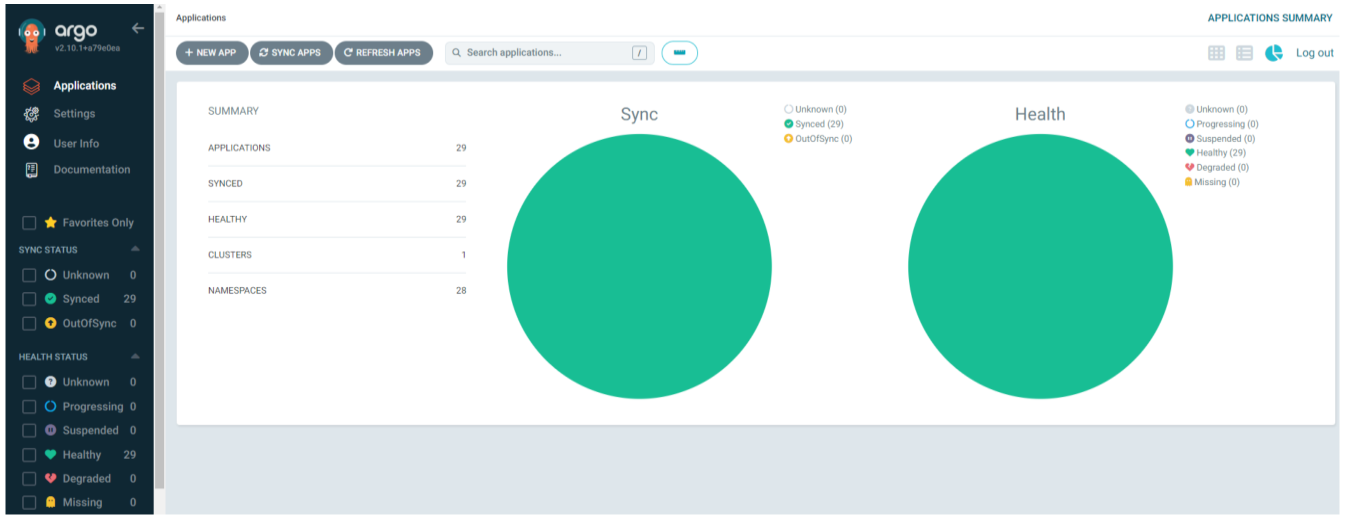
Argo CD is a declarative, GitOps continuous delivery tool for Kubernetes. It is a very convenient tool for developers, because it allows automatic synchronization between the code (desired target state) and the application (live state) with user-friendly graphics, and safe for cluster administrators, because it prevents users from having direct access to the Kubernetes API. Obviously, users will be limited by the constraints seen previously (Kyverno).

In Argo CD it is possible to define:

- Define hosted namespaces
- Black/Whitelist cluster-wide/namespaced resources
- Policy for authorized users

It also allows integration with:

- SSO (INDIGO-IAM)
- Prometheus
  - Prometheus can access metrics exported from Argo
  - Prometheus injects container metrics into the Argo dashboard



SUMMARY	
APPLICATIONS	29
SYNCED	29
HEALTHY	29
CLUSTERS	1
NAMESPACES	28

Operations

Limited external access points. Only 2 FloatingIPs were used for the entire cluster: one for the LB and the other for a bastion VM. Access via SSH to this VM is permitted only to system administrators for maintenance and management reasons: on this VM there are software such as kubectl, k9s (k8s textual UI) and Helm (used to deploy most of the software seen so far).

The ETCD backup is saved on the bastion node: it periodically collects the data from all 3 server nodes, keeping the last 5. To save data longer, these backups are taken from the node, and, thanks to the BackupPC software, they are saved on tape via TSM (IBM).

Monitoring occurs on 2 levels:

- As regards the infrastructure side, the **cluster nodes** are under control via Sensu-go, InfluxDB and Grafana, with alarms on Slack, Teams and e-mail.
- At the **cluster** and **application** level we use monitoring via Prometheus and Grafana (helm chart kube-prometheus-stack), present on the cluster itself.

Conclusions

Summary and future prospects

To summarize, we have given some tips for creating a K8s cluster in **high availability** (master redundancy, LB, decoupled storage), **quickly** (through the use of automation) and **secure** (Center for Internet Security).

Future steps:

- Finish **log collection** and indexing via Big Data Platform (Filebeats, Kafka, Logstash, OpenSearch)
- Integration with **OIDC/Oauth** for Kubernetes API access
- Investigate the use and the adoption of **backup** solutions (Ceph side)

Purposes:

- Adopt this approach as a platform for **other projects** and contexts

Ministero dell'Università e della Ricerca | Italiadomani PIANO NAZIONALE DI RIPRESA E RESILIENZA | PNC Piano nazionale per gli investimenti complementari al PNRR Ministero dell'Università e della Ricerca | INFN | DARE DIGITAL LIFELONG PREVENTION

Thank You!

This research was co-funded by the Italian Complementary National Plan PNC-I.1 "Research initiatives for innovative technologies and pathways in the health and welfare sector" D.D. 931 of 06/06/2022, "DARE - DigitAl lifelong pRevEntion" initiative, code PNC0000002, CUP: B53C22006450001