Finanziato dall'Unione europea NextGenerationEU

Ministero dell'Università e della Ricerca

Italiadomani PIANO NAZIONALE DI RIPRESA E RESILIENZA

ICSC Centro Nazionale di Ricerca in HPC, Big Data and Quantum Computing

ICSC

Centro Nazionale di Ricerca in HPC,
Big Data and Quantum Computing

Developing and testing of a flexible and scalable High Throughput Data Analysis platform

Gianluca Sabella, Bernardino Spisso on behalf of WP2.5 group

Annual Meeting Spoke2 - WP2.5 – 18/12/2023 to 20/12/2023

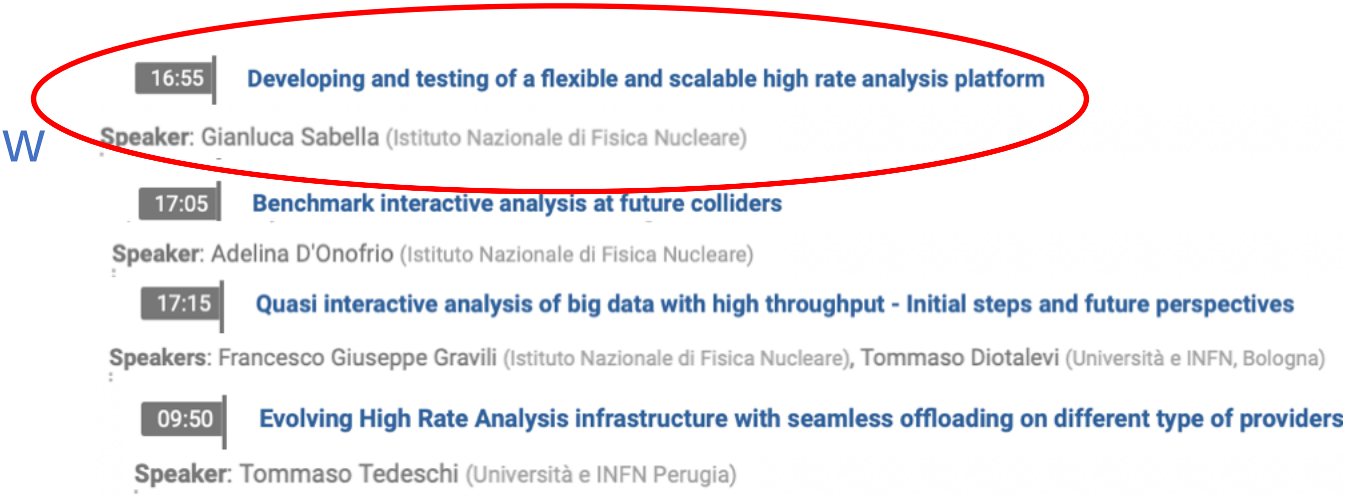
High throughput data analysis platform

Goal: to provide the users with an infrastructure that represents a tradeoff between deployment speed-flexibility, resource efficiency and service performance

Solution being tested: the use of container technology (via Docker 20.10) that runs the applications and the Kubernetes tool for orchestration

Outline of the talk

- ✓ Kubernetes test infrastructure overview
- ✓ Tools for installing and managing the Kubernetes infrastructure
- ✓ Tools and metrics adopted to test the high throughput data analysis performance



16:55 **Developing and testing of a flexible and scalable high rate analysis platform**

Speaker: Gianluca Sabella (Istituto Nazionale di Fisica Nucleare)

17:05 **Benchmark interactive analysis at future colliders**

Speaker: Adelina D'Onofrio (Istituto Nazionale di Fisica Nucleare)

17:15 **Quasi interactive analysis of big data with high throughput - Initial steps and future perspectives**

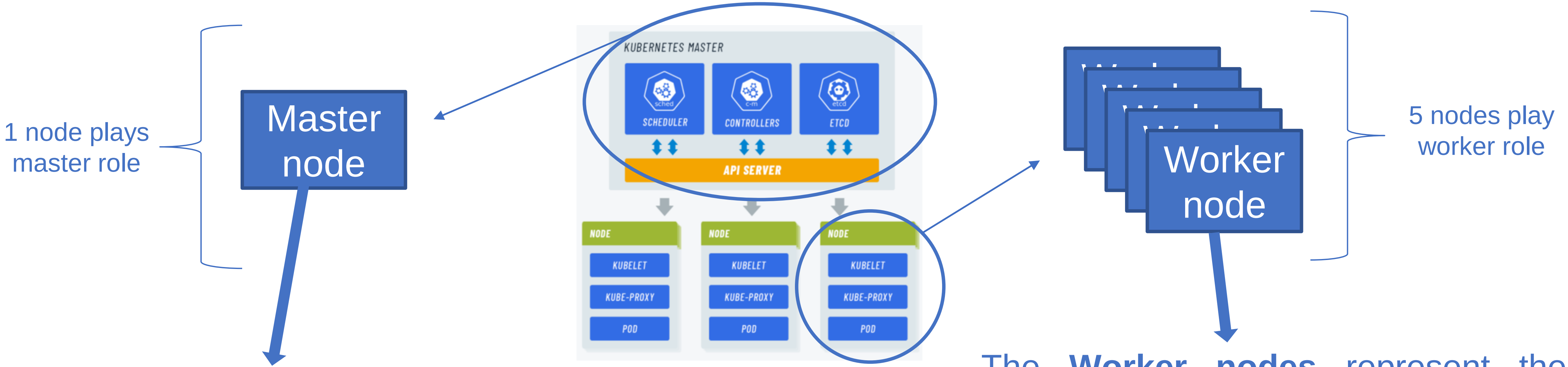
Speakers: Francesco Giuseppe Gravili (Istituto Nazionale di Fisica Nucleare), Tommaso Diotalevi (Università e INFN, Bologna)

09:50 **Evolving High Rate Analysis infrastructure with seamless offloading on different type of providers**

Speaker: Tommaso Tedeschi (Università e INFN Perugia)

Architecture of the testbed infrastructure

Local testbed infrastructure provides 6 nodes, orchestrated via **Kubernetes (1.26.3)**:



1 node plays master role

Master node

5 nodes play worker role

Worker node

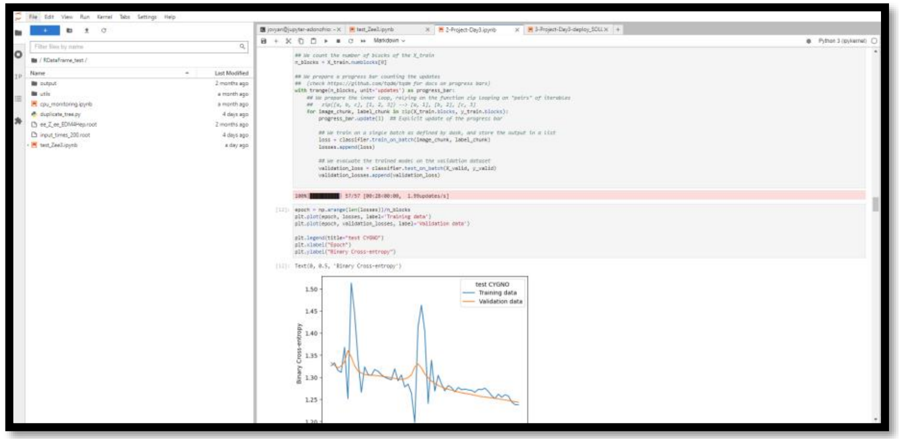
The **Master node** coordinates and manages the entire Kubernetes infrastructure. It includes the API Server, the Controller Manager, the Scheduler, and etcd.

The **Worker nodes** represent the nodes of the cluster where containers are executed. Each one hosts the Kubelet, Kube-proxy and the containers, providing the actual computational capacity of the cluster.

Tools for the high throughput data analysis

Through the Kubernetes infrastructure created, we set up a prototype to respond to previous requests, offering the user computing resources. The tools used were:

- ✓ **Jupyter Hub/Lab**: interactive, web-based development environment capable of managing multiple accesses



Name	Last Modified
output	2 months ago
utils	a month ago
cpu_monitoring.ipynb	a month ago
duplicate_tree.py	4 days ago
ee_Z_ee_EDM4Hep.root	2 months ago
input_times_200.root	4 days ago
test_Zee3.ipynb	a day ago

```
## We count the number of blocks of the X_train
n_blocks = X_train.numblocks[0]

## We prepare a progress bar counting the updates
##  (check https://github.com/tqdm/tqdm for docs on progress bars)
with trange(n_blocks, unit='updates') as progress_bar:
    ## We prepare the inner loop, relying on the function zip looping on "pairs" of iterables
    ##   zip([a, b, c], [1, 2, 3]) --> [a, 1], [b, 2], [c, 3]
    for image_chunk, label_chunk in zip(X_train.blocks, y_train.blocks):
        progress_bar.update(1)  ## Explicit update of the progress bar

        ## We train on a single batch as defined by dask, and store the output in a list
        loss = classifier.train_on_batch(image_chunk, label_chunk)
        losses.append(loss)

        ## We evaluate the trained model on the validation dataset
        validation_loss = classifier.test_on_batch(X_valid, y_valid)
        validation_losses.append(validation_loss)
```

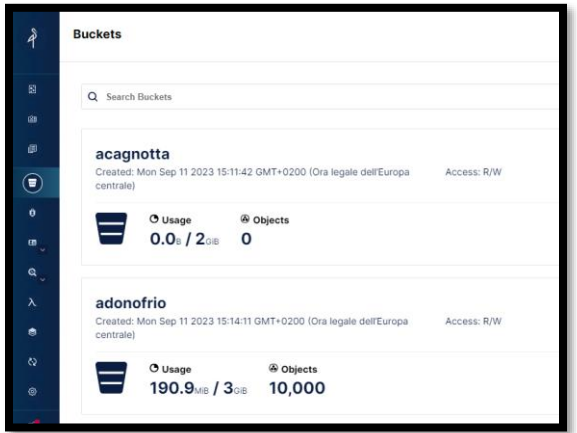
```
100%|██████████| 57/57 [00:28<00:00,  1.99updates/s]
```

```
[12]: epoch = np.arange(len(losses))/n_blocks
      plt.plot(epoch, losses, label='Training data')
      plt.plot(epoch, validation_losses, label='Validation data')

      plt.legend(title="test CYGNO")
      plt.xlabel("Epoch")
      plt.ylabel("Binary Cross-entropy")
```

```
[12]: Text(0, 0.5, 'Binary Cross-entropy')
```

- ✓ **DASK:** Open-source and flexible Python library for parallel computing that can use the most popular batch systems (SLURM, HTCondor, etc. ...) or is provided with a native scheduler if necessary (our choice)

- ✓ **S3:** Object Storage designed to be deployed anywhere: public or private cloud, bare metal infrastructure, orchestrated environments and edge infrastructures.


Buckets

Search Buckets

acagnotta
Created: Mon Sep 11 2023 15:11:42 GMT+0200 (Ora legale dell'Europa centrale)
Access: R/W
Usage 0.0B / 2GiB
Objects 0

adonofrio
Created: Mon Sep 11 2023 15:14:11 GMT+0200 (Ora legale dell'Europa centrale)
Access: R/W
Usage 190.9MiB / 3GiB
Objects 10,000

User experience

We provided support to supply users with the tools they needed (ROOT, editors, etc. …)

The infrastructure was tested using several realistic high energy physics workflows:

- The overall execution time is the metric chosen to evaluate and compare the performance of the standard "serial" model and the parallel approach to data analysis, based on the new infrastructure

- Preliminary feasibility studies show that the execution time improves significantly exploiting the DASK distributed approach on the analysis platform

More in Adele's talk

Conclusions and Outlooks

In conclusion, we are working to provide computing services to users in a rapid and scalable way and ensure the portability of applications between different environments and their management on different infrastructures.

Next steps:

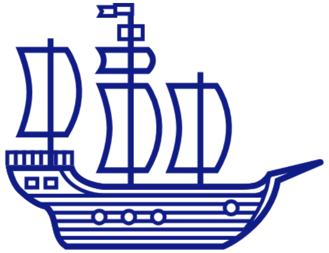
- In this prototypal development the recipe is constantly improving
- The near-term goal of the activity is to automate the high throughput data analysis deployment exploiting the ICSC computing resources

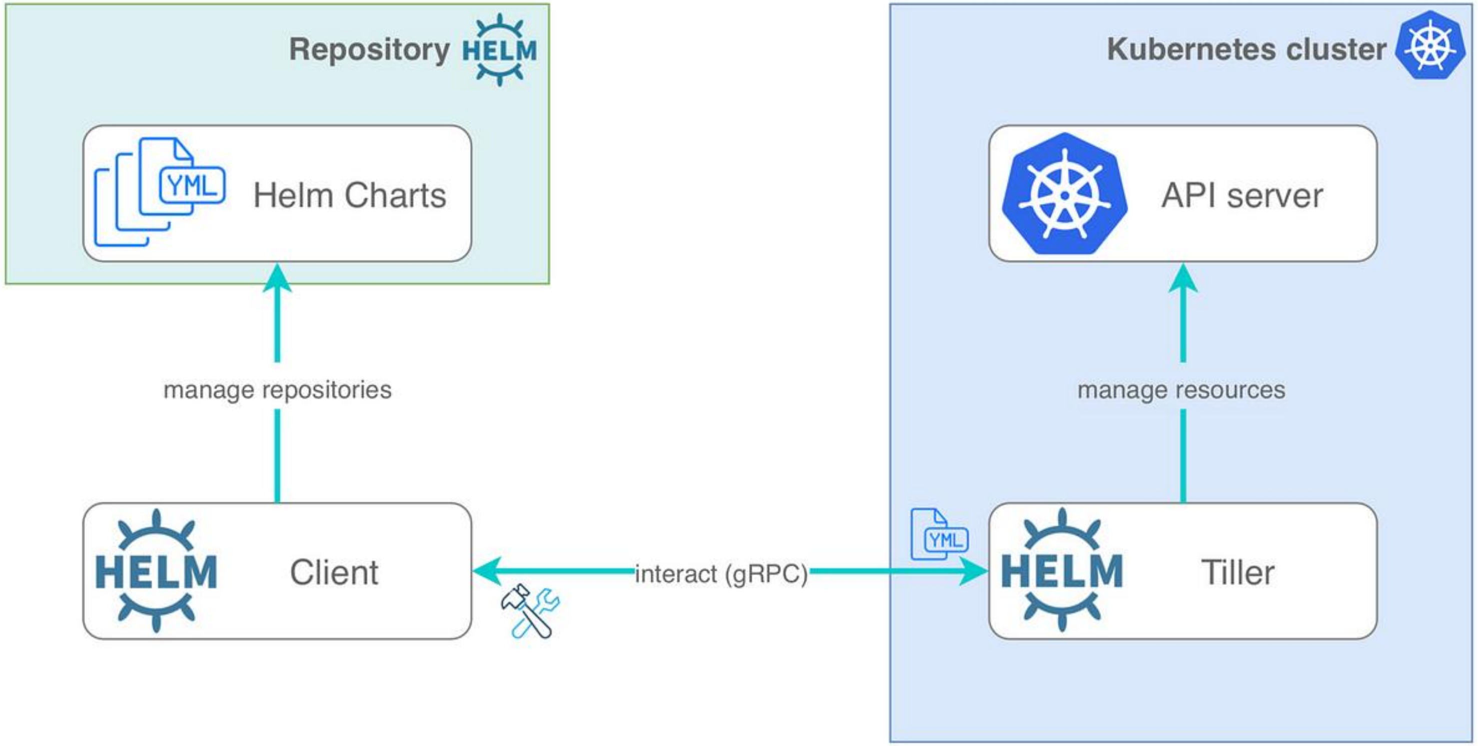
Thank you for your attention

Backup

Tools used (1): HELM Client - The package manager for Kubernetes

Helm is a packet manager that facilitates the management of Kubernetes applications by defining, installing and managing packages called charts.





Helm charts are pre-configured packages that contain all the resources needed to run an application on Kubernetes, including services, deployments, inputs, configurations, variables, etc. …

Tools used (2): Rancher Kubernetes Engine (RKE)

RKE

RKE is a certified Kubernetes distribution that runs entirely inside Docker containers.

It addresses the complexity of installing with Kubernetes by removing most host dependencies and presenting a stable path for deployment, updates, and rollbacks.

Requirement: use of a tool for interacting with the Kubernetes cluster → **The Kubectl command line tool**

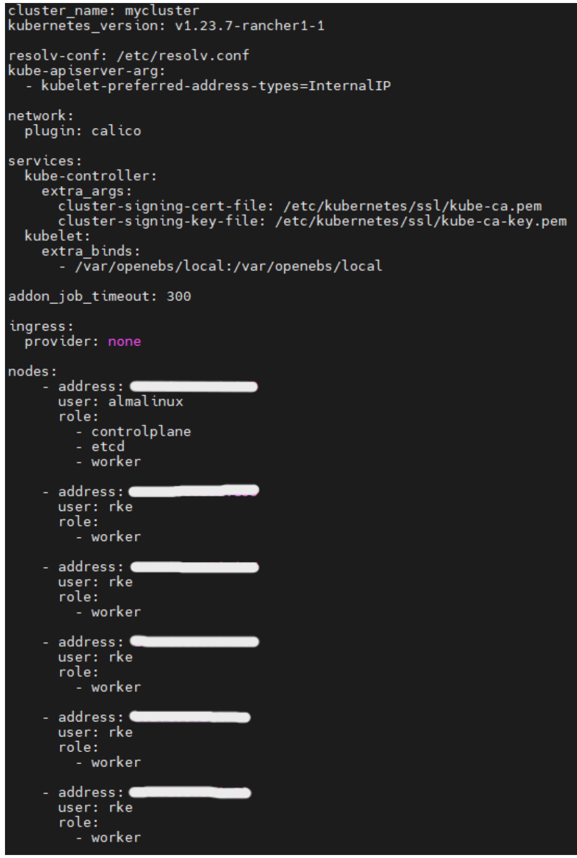
Kubernetes infrastructure generation: RKE + YAML

RKE + YAML

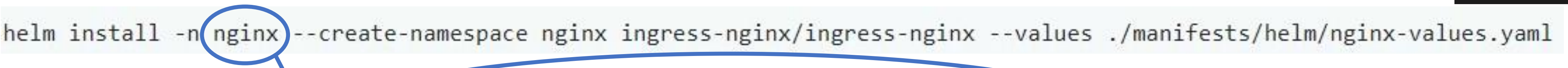
YAML is a plain-readable programming language for data serialization that is often used for writing configuration files.

The configuration of the Kubernetes infrastructure setup via YAML file (.yml or .yaml), used by RKE for cluster generation (**$ rke up**)

HELM charts are also in yaml format



```
cluster_name: mycluster
kubernetes_version: v1.23.7-rancher1-1

resolv-conf: /etc/resolv.conf
kube-apiserver-arg:
  - kubelet-preferred-address-types=InternalIP

network:
  plugin: calico

services:
  kube-controller:
    extra_args:
      cluster-signing-cert-file: /etc/kubernetes/ssl/kube-ca.pem
      cluster-signing-key-file: /etc/kubernetes/ssl/kube-ca-key.pem
  kubelet:
    extra_binds:
      - /var/openebs/local:/var/openebs/local

addon_job_timeout: 300

ingress:
  provider: none

nodes:
    - address: [redacted]
      user: almalinux
      role:
        - controlplane
        - etcd
        - worker

    - address: [redacted]
      user: rke
      role:
        - worker

    - address: [redacted]
      user: rke
      role:
        - worker

    - address: [redacted]
      user: rke
      role:
        - worker

    - address: [redacted]
      user: rke
      role:
        - worker

    - address: [redacted]
      user: rke
      role:
        - worker
```



```
helm install -n nginx --create-namespace nginx ingress-nginx/ingress-nginx --values ./manifests/helm/nginx-values.yaml
```

NGINX: Open-source web server also used as a reverse proxy, HTTP cache, and load balancer.