The DLaaS behind the scenes

VRE working group monthly meeting

Alba Vendrell Moya and Elena Gazzarrini

Technologies: GitOps

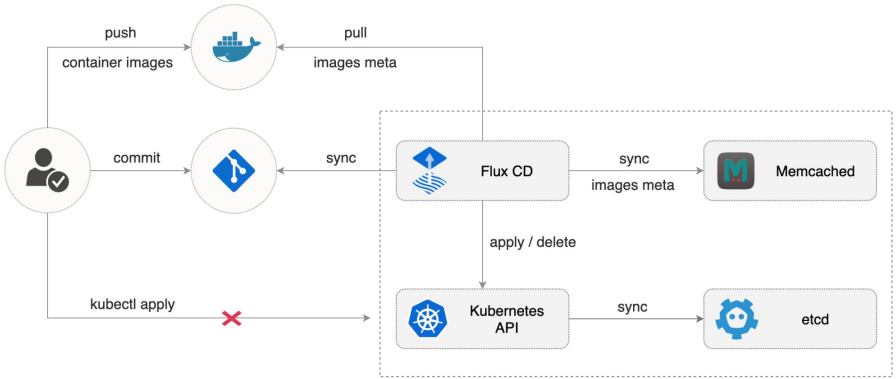
Containers → **Docker**
Think about the container as a machine

Container Orchestration → **K8s** (**Flux/Helm**, CI/CD)
To organise multiple containers, in a cluster

Networking: HTTP Server → **Apache HTTP** Server (httpd)

Authorisation/Authentication → **X509/OIDC, TLS/SSL**

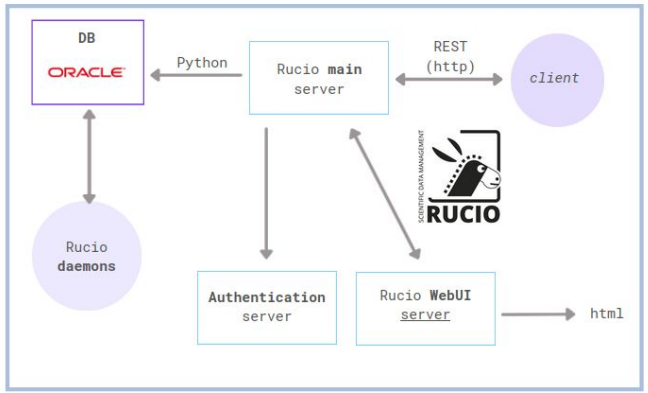
DataBase → **Relational DB** (Oracle, Postgres)
With ACID (Atomicity, Consistency, Isolation and Durability) characteristics

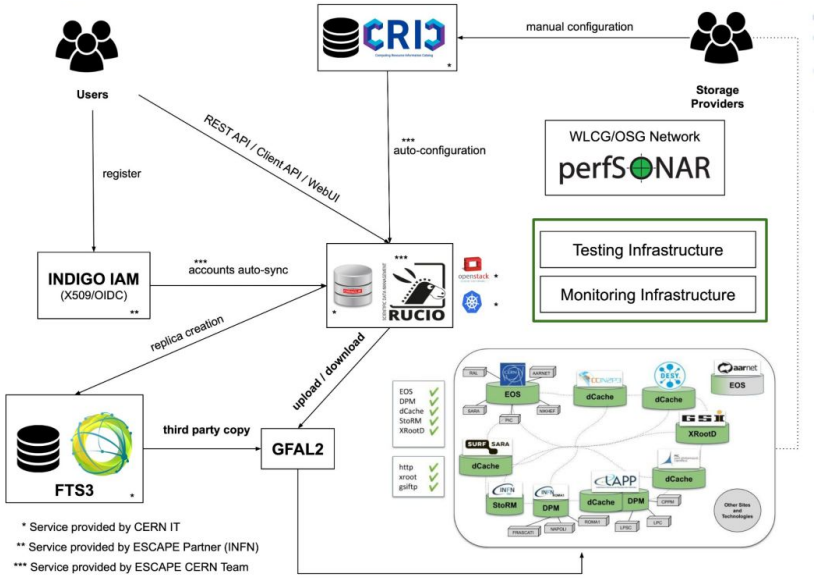


Main infrastructure - ESCAPE Rucio instance

The ESCAPE infrastructure on which the DLaaS sits on is composed of:

- **Main Server**
  - handles REST requests to the resources (Apache HTTP Server)
- **Authorization Server**
  - handles REST authentication/authorization requests (Apache HTTP Server)
- **WebUI Server**
  - Rucio GUI (Apache HTTP Server)
- **Daemons**
  - Python modules that interact with the DB





Cluster set-up: step by step guide

- Cluster creation on Openstack (magnum)
- Secrets management with Mozilla SOPS
- Network
- RSE (storage) management and configuration
- Monitoring
- Helm installing server, daemons, webui
- Jupyter notebook + rucio jupyterlab extension

The K8s cluster + secrets

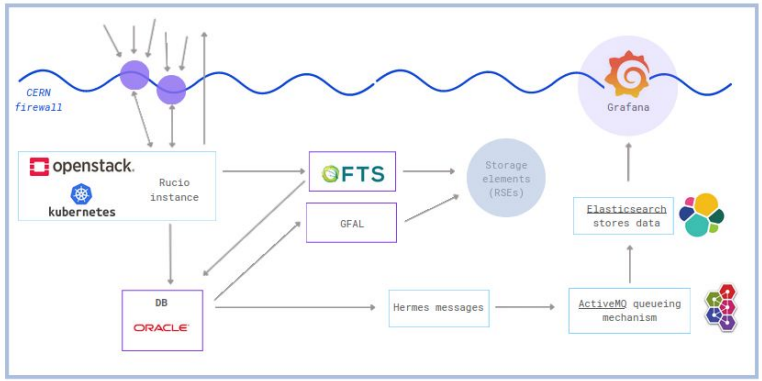
Nodes(all)[8]

NAME↑	STATUS	ROLE	VERSION	PODS	CPU	MEM
eosc-cluster-cqh2glh7cswm-master-0	Ready	master	v1.21.1	7	207	2352
eosc-cluster-cqh2glh7cswm-master-1	Ready	master	v1.21.1	7	201	2453
eosc-cluster-cqh2glh7cswm-master-2	Ready	master	v1.21.1	7	192	2642
eosc-cluster-cqh2glh7cswm-node-0	Ready	<none>	v1.21.1	12	248	4512
eosc-cluster-cqh2glh7cswm-node-1	Ready	<none>	v1.21.1	11	157	4218
eosc-cluster-cqh2glh7cswm-node-2	Ready	<none>	v1.21.1	16	160	3791
eosc-cluster-cqh2glh7cswm-node-3	Ready	<none>	v1.21.1	19	138	4178
eosc-cluster-cqh2glh7cswm-node-4	Ready	<none>	v1.21.1	10	113	4389

- 3 master nodes, 5 worker nodes, cluster creation with openstack magnum (service at CERN which automates many steps)
- Install **flux**
- Create github public repository
- Bootstrap flux on it
- Rucio expects secrets (certificates, DB passwords, OIDC client ID, etc.) before starting the service
  - .p12 certificates, split into host and key files
  - gridCA certificates
  - TLS certificates
  - client_id and client_secret of the Rucio Admin account created with the Identity Provider (ESCAPE IAM for us) → needed for JSON web tokens (JWTs) and OAuth2.0 authentication and authorization with Rucio
  - Database credentials (Oracle, PostgreSQL, MySQL/MariaDB are currently supported)
- Follow **SOPS** tutorial to automatically apply encrypted secrets in cluster once the .yaml file is pushed to repository (without the need of doing '*kubectl apply*') as a public key is shared in the repo

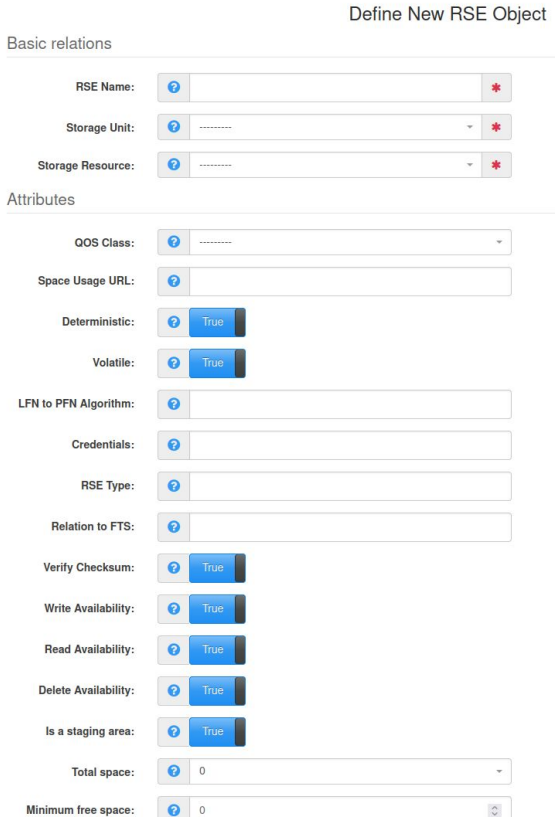
Network

- 2 nodes of the cluster are set as K8s **Ingress controllers**
  - They accept traffic from outside, and **load balance** it to pods (containers) running inside the platform
- Set **NGINX** as Ingress controller, as it is most popular and open source way to have a *reverse proxy* (to protect the server) + supports X509
- External traffic
  - **LanDB**-alias is set for eosc-auth.cern.ch, eosc-main.cern.ch, eosc-webui.cern.ch (by default, the CERN outer perimeter firewall blocks incoming access to systems on the CERN site ➜ need to request to open for the lanDB-alias property configuration)
- CERN Openstack offers a [Load Balancing as a Service] that we are checking out



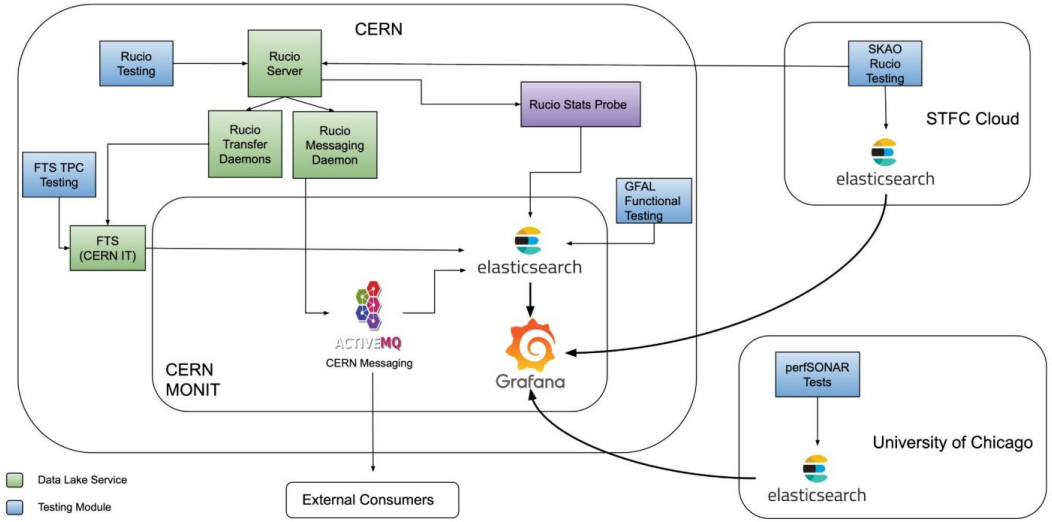
RSE configuration + CRIC

- RSEs can be configured
  - manually through Rucio commands
  - In **CRIC**
    - easier RSE management
    - script to sync the service with new RSE creation via JSON request
- Plan to have one in EULAKE-1 (still testing token functionalities there)
- Would any other institute provide us with at least two other RSEs?



Monitoring

- Logging producer is requested at cluster creation
- Logs are injected into **Grafana** for monitoring dashboards
- We will add some extra cluster logs
- FTS service is already configured to push data into CERN Monit



Rucio Helm releases

- The rucio helm charts can be found in the rucio repo
- Each Helm Release will start a deployment of each of:
  - *Server*
  - *Webui*
  - *Daemons* - figuring out minimal ones needed to have the service running
- Each pod will spawn a container with the Rucio configuration inside
- Secrets need to be already applied to the cluster

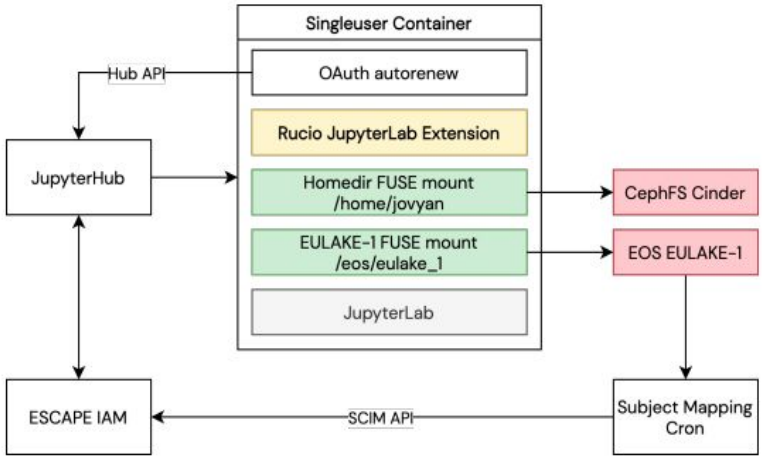
DLaas
Feature Highlights

The goal of the service is to abstract the complexities of the Data Lake from the scientists.
This way, scientists can focus their time on doing science instead of data procurement.

- Multiple notebook environment selection
- Rucio data browser (with scope browser and wildcard search)
- “Add to shopping cart” for data catalogue
  - DID is attached as a metadata in the Notebook file
- Injects a variable containing local file path, ready to be used
- Direct file upload to Rucio
- Scratch space for large files (EOS FUSE mount)

Deployment

- Deployed in Kubernetes @ CERN Openstack, using Zero-to-JupyterHub Helm chart.
  - https://escape-notebook.cern.ch
- CI/CD
  - Gitlab CI - Container build
  - Flux2 - Kubernetes manifest
- OAuth authentication using ESCAPE IAM.
- Uses Rucio JupyterLab Extension in Replica mode
  - Connected to ESCAPE Data Lake (escaperucio.cern.ch; **rucio_host**)
  - Automatically preconfigured to use OIDC authentication (**RUCIO_DEFAULT_AUTH_TYPE**)
  - Has a FUSE mount to EULAKE-1 RSE (EOS; **RUCIO_RSE_MOUNT_PATH**)
  - Making files available means creating a replication rule to move files to **EULAKE-1** (**RUCIO_DESTINATION_RSE**)

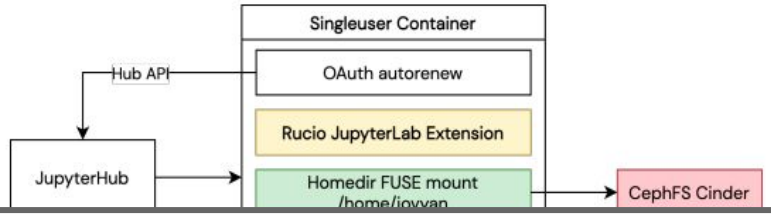


(*) **CONFIGS**

Deployment

- Deployed in Kubernetes @ CERN Openstack, using Zero-to-JupyterHub Helm chart.
  - https://escape-notebook.cern.ch
- CI/CD
  - Gitlab CI - Container build
  - Flux2 - Kubernetes manifest
- OAuth authentication u
- Uses Rucio JupyterLab
  - Connected to ESCAP
  - Automatically precon (**RUCIO_DEFAULT_A**
  - Has a FUSE mount t **RUCIO_RSE_MOUNT**
  - Making files available to **EULAKE-1** (**RUCI**

(*) **CONFIGS**

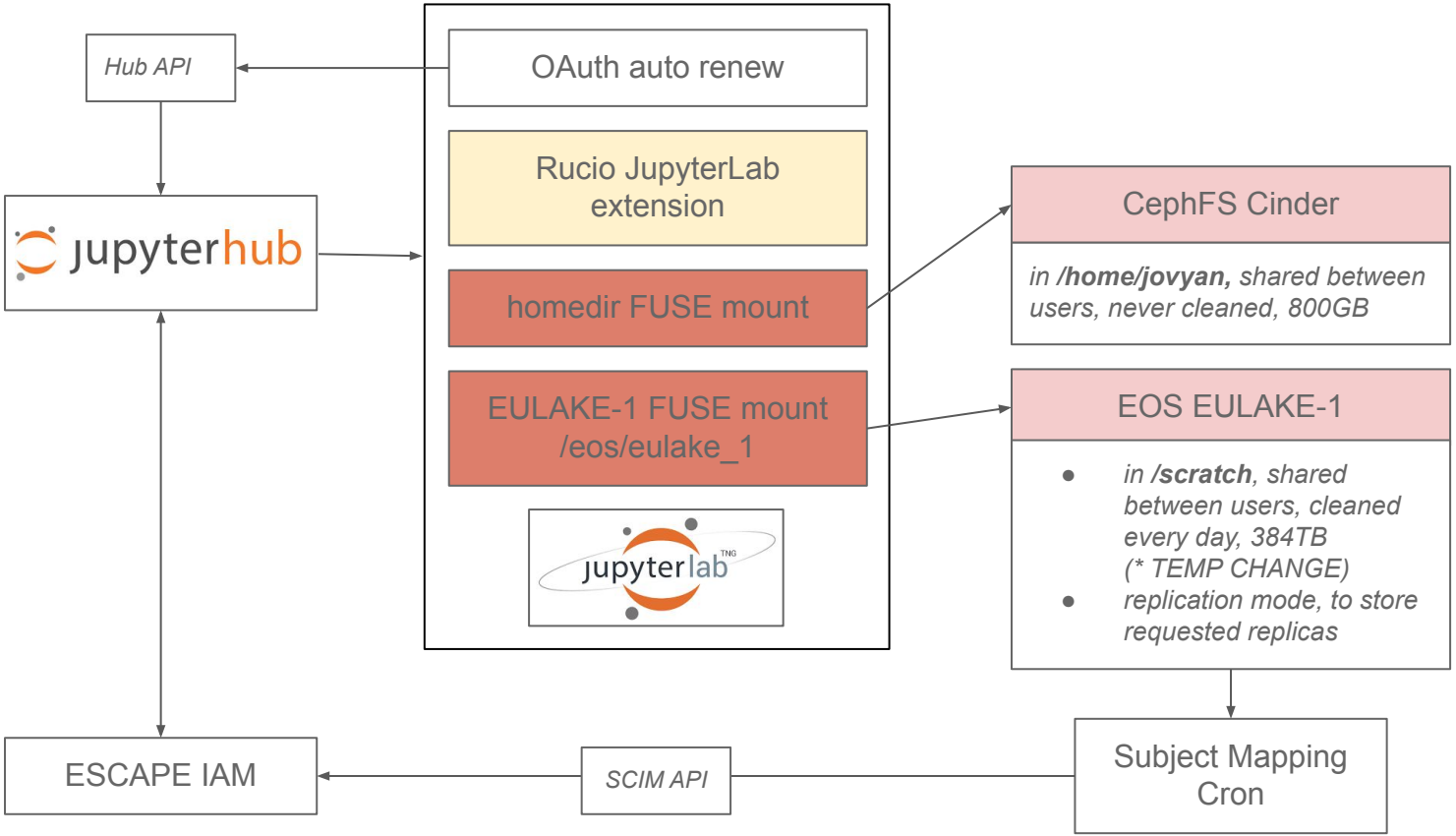


Replica Mode

In this mode, the files are transferred by Rucio to a storage mounted to the JupyterLab server. In order to use the extension in this mode, you need to have the following set up:

- A JupyterLab version 2 installation.
- At least one Rucio instance.
- A storage system that is attached to the JupyterLab installation via FUSE.
  - The storage system should be compatible with Rucio and added as a Rucio Storage Element.
  - The storage element will be shared among multiple users, so be sure to allow all users who will be using the extension to have read permission to the path.
  - It's recommended that quotas be disabled, since the extension does not care if the replication fails because of quota error.

DLaaS current status



FUSE mount to EOS eulake

- There are two FUSE mounts to the same EOS instance:
  - /eos/eulake_1 → /eos/eulake/tests/rucio_test/eulake_1
  - /scratch → /eos/eulake/tests/jupyter-scratch
- FUSE mount is implemented using k8s DaemonSet, mounting to a folder in the host, with Bidirectional mount propagation
- Singleuser containers bind to the mount folder, with **HostToContainer** mount propagation
- Uses OAuth2 authentication
  - ESCAPE IAM user is mapped to EOS user using crons

OAuth2 in EOS FUSE mount

- In the singleuser container:
  - JWT is stored in a file in the following format:

    ```
    oauth2:<jwt>:<token-introspection-endpoint>
    Example: oauth2:eyJ...:iam-escape.cloud.cnaf.infn.it/userinfo
    ```

    - Note: token introspection endpoint doesn't have the "https://" part
    - The token file must have at most 0600 permission
  - An environment variable needs to be set:
    - OAUTH2_TOKEN=FILE:/path/to/token/file
  - In the EOSFUSE DaemonSet container:
    - EOS FUSEx daemon (eosxd) needs to be configured for SSS authentication
    - SSS keytab needs to be present

- (*) https://eos-docs.web.cern.ch/using/oauth2.html

Singleuser container setup

- OAuth token exchange (eos-eulake and Rucio)
  - Modified version of SWAN's KeyCloakAuthenticator
- Enable token autorenewal
  - Uses SwanOauthRenew
- Write token files to /tmp
- Set OAUTH2_TOKEN env for EOS authentication
- Write rucio.cfg file

Alternative Rucio instances - inspiration

- SKAO: https://gitlab.com/ska-telescope/src/ska-rucio-prototype
- ATLAS: https://gitlab.cern.ch/atlas-adc-ddm/rucio-k8s-setup/-/blob/master/README.md#L48-118
- Our DL: https://indico.cern.ch/event/1069544/contributions/4497649/attachments/2311244/3933259/Data%20Lake%20as%20a%20Service%20for%20Open%20Science.pdf
- ASTRON replicating DLaaS: https://git.astron.nl/groups/astron-sdc/escape-wp5/-/wikis/Meeting-Notes/Spring-2022-Busy-Week/Replicating-Data-Lake-as-a-Service

Reading list

- **The ESCAPE Data Lake: The machinery behind testing, monitoring and supporting a unified federated storage infrastructure of the exabyte-scale** https://www.epj-conferences.org/articles/epjconf/abs/2021/05/epjconf_chep2021_02060/epjconf_chep2021_02060.html
- **ESCAPE Data Lake: Next-generation management of cross-discipline Exabyte-scale scientific data** https://www.epj-conferences.org/articles/epjconf/abs/2021/05/epjconf_chep2021_02056/epjconf_chep2021_02056.html