A distributed data-mining software platform for extreme data across the compute continuum

D4.3 Final Release of the Compute Continuum and Final Integration Plan

Version 1.0

Documentation Information

Contract Number	**101093110**
Project Website	https://www.extract-project.eu
Contractual Deadline	M30, 30th June 2025
Dissemination Level	Public
Nature	Other
Author	IBM Research - Israel (IBM)
Contributors	BSC, IKL, SIX, BIN
Reviewer	IKL
Keywords	Data, integration, CCS, continuum, SkyStore, multi-cloud, multi-premise

Change Log

Version	Description Change
V0	Initial draft of the table of contents and collaboration assignments
V0.1	Initial IBM contributions
V0.2	OBS contributions
V0.4	BIN contributions
V0.5	Review (IKL)
V0.6	Apply comments from review
V0.7	Final revision (IBM)
V1.0	Ready to submit

The EXTRACT Project has received funding from the European Union's Horizon Europe programme under grant agreement number 101093110.

Table of Contents

1. Introduction

This document marks the final release of the Compute Continuum and Final Integration Plan developed within Work Package 4 (WP4) of EXTRACT. Complementing the first release documented in Deliverable D4.2, this final version represents implementation and refinement of the original design. Consequently, this deliverable presents short recaps of the original design followed by reports of only significant changes compared to D4.2, as following:

- Multi-premise architecture - new deployments
- Enhancements to the data backbone – SkyStore
- Security enhancements
- Validation (Demonstrators)

1.1. Structure

This document is organized in 8 sections: 6 content sections + abbreviations and references:

- Section 1 introduces the document and gives a main view of the structure of the document.
- Section 2 recalls the multi-premise architecture of EXTRACT and discusses new deployments
- Section 3 recalls the data backbone of EXTRACT and discusses recent changes
- Section 4 discusses security work of the recent period
- Section 5 lists relevant demonstrators validating the recent work
- Section 6 concludes this deliverable and discusses future directions

1.2. Relationship with other WPs and Deliverables

Deliverable	Tasks	Relation
D1.3	T1.3	SkyStore enhancements have been used in the implementation of PER, and being considered for a multi-cloud deployment of TASKA-C and TASKA-D. A joint demonstrator WP1/WP2/WP4 in D1.3 is referenced from this document.
D2.3	T2.3	The final release of the Data-Mining Framework relies on proper integration of the data backbone and data catalog, delivered at this stage.
D3.2/D3.3	T3.2, T3.3	The multi-premise (multi-cloud) orchestration correlates side-by-side with multi-premise architecture (Compute Continuum Site – CCS) and multi-premise data backbone (SkyStore) to deliver EXTRACT's vision of an efficient, global data-driven computation.

D4.2	T4.2, T4.3, T4.5	This deliverable depicts the final version of the compute continuum abstraction layer, finalizing the development work of these tasks, bringing together the service-based interoperability (T4.2), security (T4.3) and software integration (T4.5) to a successful completion.

Table 1. Relationship with other WPs

2. Final Multi-Premise Architecture

The multi-premise EXTRACT architecture that was presented in D4.2 has been proven effective using both use-cases and as such remains unchanged to the end of the project. In this section we first present a brief overview of the architecture. Next, we discuss work done on this topic in the milestone period, which mostly involves extending the installed base of EXTRACT to new premises.

2.1. Architecture Overview - Recap

As explained in D4.2, EXTRACT addresses a problem of distributing computation along a compute continuum, from edge to cloud. This implies architectural aspects such as deployment elasticity, security, data propagation and access, and interoperability across different premises (edge, HPC, cloud), each of which can have multiple different instances, e.g., multiple clouds, multiple HPC data-centers, etc.

The selected architecture solution defines each EXTRACT instance as a linked non-empty set of **CCS** - Compute Continuum Sites. See Figure 1 below. Each CCS is essentially a Kubernetes [1] cluster with EXTRACT services inside, thus providing a uniform environment for applications based on EXTRACT, regardless of the surrounding premise - edge, cloud, HPC (can be multiple premises of each type). Each EXTRACT instance comprises a set of CCS, each deployed in a premise. The actual set of EXTRACT services in a CCS varies according to the application requirements with details in D4.2.

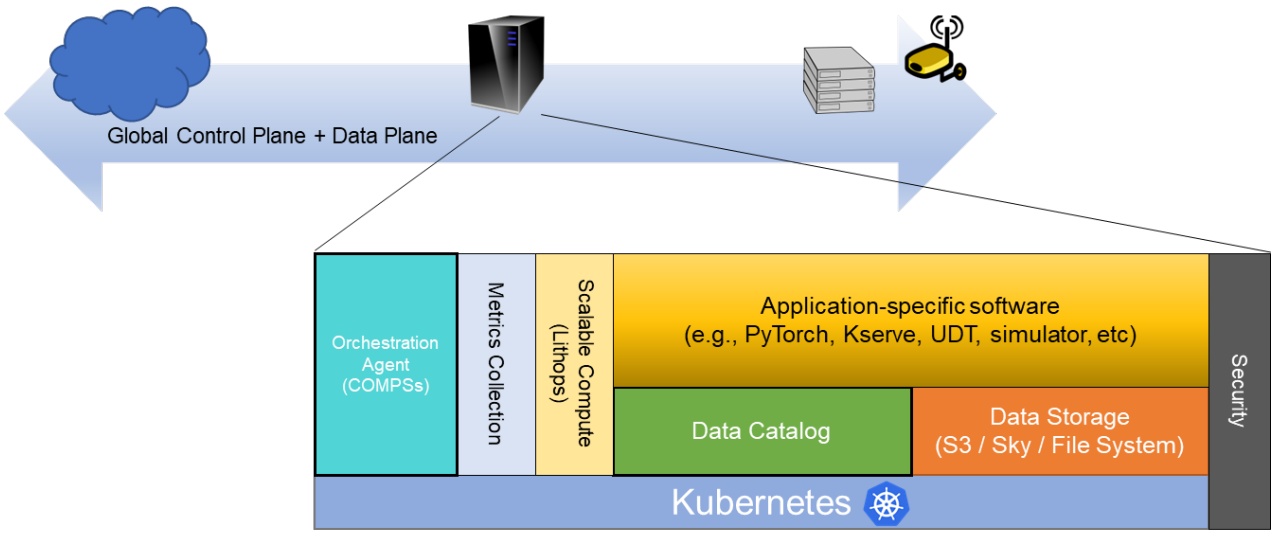

Figure 1. EXTRACT Multi-Premise Architecture

All the CCS of the same EXTRACT instance are linked together such that EXTRACT applications can be seamlessly and automatically deployed and managed across multiple premises, with data propagated to application components at each CCS as needed. Thus, the global cross-CCS backbone consists of data and orchestration (control). The data backbone is implemented using SkyStore and addressed in Section 3 of this document. The orchestration backbone is implemented using COMPSs and is discussed in D3.2 and D3.3.

2.2. Recent Deployments

During the recent work period, there have been additional successful deployments of EXTRACT, as described below. These deployments consist of two public cloud providers available for academic users: EGI and EOSC, plus an edge cloud at Paris Observatory. Each of these deployments is based on a single CCS with independent S3 storage.

EGI

EGI (European Grid Infrastructure) [2] is a federation of computing and storage resource providers united by a mission to support research and development. CESNET (Czech Education and Scientific NETwork) together with e-INFRA CZ provide for EGI a Cloud Container Platform based on the Rancher [3] software.

Rancher provides a Platform-as-a-Service with Kubernetes APIs and a web portal (https://rancher.cloud.e-infra.cz/dashboard/). Authentication is granted through the Edugain identity federation.

S3 file service

Besides the Kubernetes environment, Rancher provides many prepackaged apps in the form of Helm [4] charts. We used this functionality to create a S3 file service based on MinIO [5] using a K8s Persistent Volume Claim [6]. During the tests, two customizations were needed: to handle big files, we added the annotation nginx.ingress.kubernetes.io/proxy-body-size : 0 to all ingresses, and we later rebuilt the docker image with a newer version for a better S3 compatibility.

We also tested the S3 file service from OVH based on Ceph [7].

Kubernetes service

The portal gives access to a Kubernetes config file. We first tested the service with a simple pod without any issue.

Test environment

We used TASKA use case C to test the EXTRACT architecture. The script used is a sample pipeline with rebinning, calibration and imaging steps, orchestrated with COMPSs.

We have encountered a problem with job submission by Lithops [8]: Rancher expects jobs to meet security requirements and there is no way to set them with lithops' public API. We managed to get around the problem by modifying the JOB_DEFAULT variable in the file serverless/backends/k8s/config.py with the lines:

```
securityContext:
  allowPrivilegeEscalation: false
  runAsNonRoot: true
  runAsUser: 1000
  capabilities:
    drop: ["ALL"]
  seccompProfile:
    type: "RuntimeDefault"
```

An issue has been created in lithops' github.

As the calculation code used in TASKA use case C is architecture-dependent, we had to prepare the docker image on a machine using the same CPU as the cloud platform.

EOSC

EOSC (European Open Science Cloud) [9] is a research infrastructure funded by the European Commission. EOSC members can access many services, such as File Sync & Share, Interactive Notebooks, Large File Transfer, Virtual Machines and a Cloud Container Platform. The latest is a Kubernetes platform based on OKD [10], the community version of RedHat OpenShift [11]. The authentication is also granted through the eduGAIN [12] identity federation.

S3 file service

We reused our MinIO service on the EGI platform for this test.

Kubernetes service

After logging in to the portal, the top-right menu generates an authentication token and provides a login command that creates the Kubernetes configuration file.

NB: The token expires in 24h. This limitation should be taken into account to build a permanent service.

Test environment

We used the same pipeline as in the EGI section. We didn't need to patch lithops for it to work on OKD, as it doesn't require any security context to run containers.

Edge cloud

Paris Observatory has recently built a private instance of OKD and a CEPH server accessible with S3 [13] and CephFS [14] protocols. Both are currently being tested and will go into production shortly.

We conducted the same tests on this platform as in the previous sections, using both the MinIO server on the EGI platform (to check that the private cloud could reliably access external data) and the local CEPH S3. In each case, we got the same satisfactory results as in the case of EOSC.

3. Data Backbone

3.1. Architecture - Recap

As explained in D4.2, the data backbone of EXTRACT is intended to connect multiple CCS to data. Per earlier design decisions, already introduced in D4.1, the core data technology for hosting and sharing data in EXTRACT is S3. That being said, S3 clusters can be distributed across the continuum just like CCS themselves, from edge deployment like a small-scale Minio, to large clusters with PBs of storage in on-prem data centers or in cloud region storage. On the other hand, EXTRACT applications should be able to access that data seamlessly and efficiently through their client endpoints, regardless of its location or distance.

In light of that, EXTRACT data backbone was chosen to be **SkyStore** (see D4.2) [15]. SkyStore is a joint open-source project of IBM Research and UC Berkeley. It is intended to enable uniform, efficient and seamless access to S3 data that is distributed across different concrete S3 clusters. As SkyStore architecture entails (see Figure 2 below), uniform access from a single endpoint is obtained through an S3 proxy that the client connects to and that can delegate the access to different concrete S3 services.

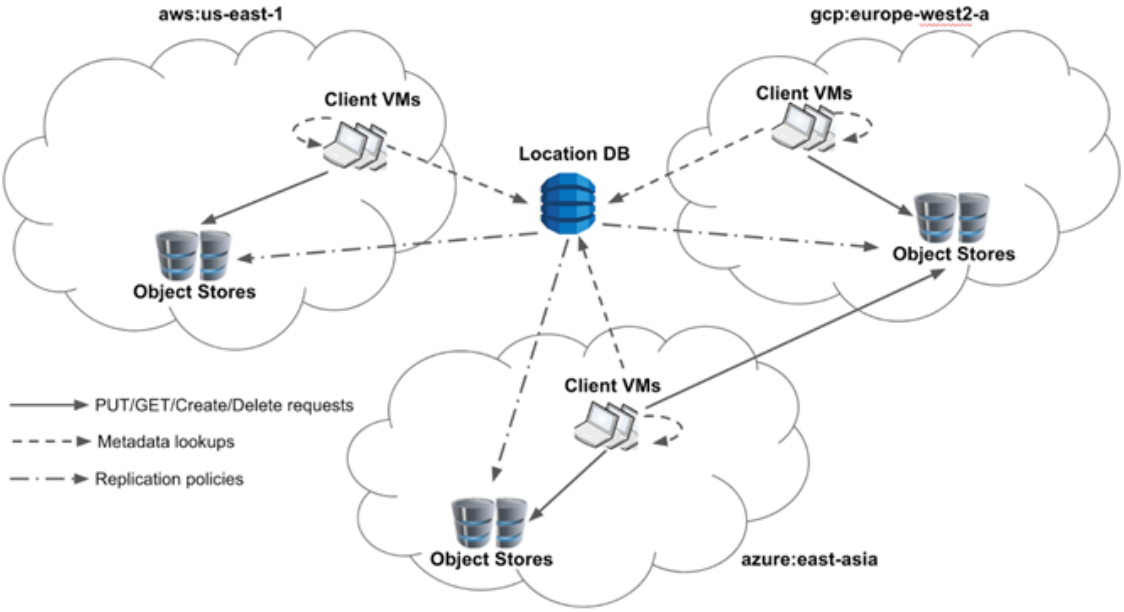

Figure 2. SkyStore Multi-Premise S3

Access efficiency (depending on application requirements) can be obtained by configuring data access and replication policies that are enforced at each S3 proxy transparently to client requests. One common example for such a policy is caching distant objects in an S3 cluster that is designated as the "local" S3 cluster for a given S3 proxy, typically for being close to the proxy network-wise. All S3-proxies connect together through a central metadata ("Location DB") server, thereby sharing the knowledge of their assigned local S3 clusters and making the sum behave as a single large S3 service, accessible to clients through the proxies. The objects are scattered over the different S3 clusters, but are bundled together in virtual buckets managed by the metadata server.

3.2. Recent Changes

3.2.1. Kubernetes Onboarding

A major effort has been invested in this milestone period in onboarding SkyStore onto Kubernetes and enhancing it in the process. SkyStore was originally a process-only solution. However, EXTRACT architecture, as explained earlier, is built on Kubernetes - pods, services, etc. Hence, the required adaptation. This work started as a quick prototype in order to do a demonstration in the mid-project review and then was extended in a more orderly fashion with testing, automation fixes, SkyStore code enhancements, detailed technical documentation, etc. In this Section we provide a technical overview of this work.

Largely, this effort consisted of 3 main stages: Containerization, Docker enablement and Kubernetes (K8s) enablement. Finally, there was a refinement stage that improved some aspects of SkyStore running on K8s and added documentation. See the details below.

Containerization

This stage involved creating Docker [16] container images that provide the main SkyStore components, namely the metadata server and the S3-proxy. Both components originate from building the same codebase - the SkyStore repository. Thus, to make development consistent, we decided to create a base image that contains the basic OS with Rust and Python and SkyStore deployed and built. Then, the image for each of the components would be built on the base image, adding only a thin layer of automation required for enabling the service of that specific component. See Figure 3 below.

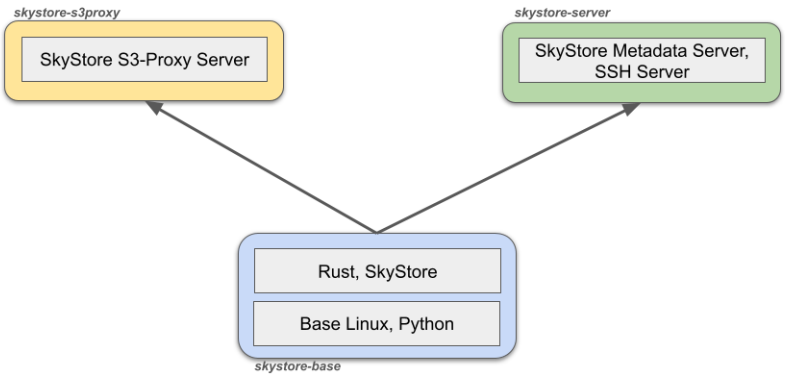

Figure 3. SkyStore Images

A first set of automation (scripts) was developed for the container service endpoints, embedded in the images. The S3-proxy service was based on the original SkyStore service. The metadata service was actually converted to an SSH server operating on a per-deployment keypair. This is a direct lesson from early experiments with SkyStore, which showed that an SSH tunnel was both an easy and secure method for distributing the original process-only version across multiple machines, as was demonstrated in D4.2. Thus, each S3-proxy connects to the metadata server as an SSH client using the private part of the SSH keypair, whereas the metadata server has the public part of the keypair.

A second set of automation was developed for building, pushing/pulling and tagging/untagging the set of 3 images as a single operation. Finally, a third set of automation was developed to handle the configuration of the components as Docker env files: SkyStore-specific configuration, SSH keys, S3 configuration for the "local" S3. The last two automation sets proved later to be extremely useful in developing the next stages.

Docker Enablement

A reasonable testing ground for any complex service layout in container form is a single-machine Docker deployment, before adding the extra complexities of Kubernetes and distribution. Hence, this was the next step. Another set of automation was developed for running a metadata server (aka "SkyStore Service") with one or more S3 proxies connected to it. Each S3 proxy can possibly connect to a different "local" S3 cluster. All this is run on a single Docker machine. See Figure 4 for the resulting layout.

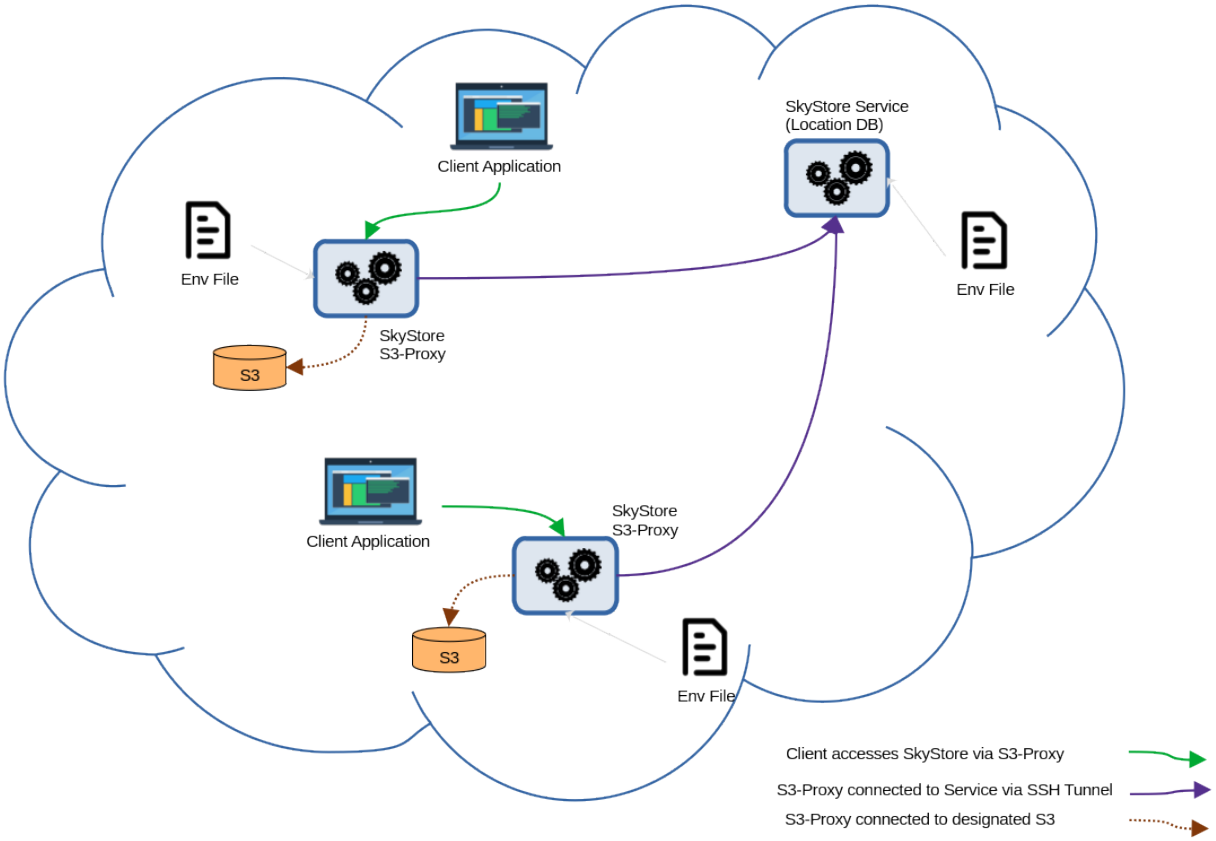

Figure 4. SkyStore in Docker

In this form, the service can already be tested using the aws CLI, using a test scenario similar to the D4.2 demonstrator, where we connect to one S3 proxy to operate on buckets and objects and then observe the changes from a different S3 proxy (possibly after some propagation time - recall that even S3 guarantees only eventual consistency between different endpoints). This gives the illusion of a single S3 system.

This setup could be easily extended to a multi-machine and multi-cluster setup using e.g., Docker Swarm [17], but this direction seems to be of little interest either to EXTRACT or to the public in general, because of the much more significant dominance of Kubernetes, so it was not pursued. Docker is used primarily for testing.

Kubernetes Enablement

And so, we arrived at the target platform of Kubernetes. In terms of components and connectivity, it is the same as in Docker and as in the process-only version. There are several important differences, though, mostly from incorporating K8s capabilities into the architecture. The final layout is described in Figure 5. We use standard Kubernetes symbols to denote the various Kinds of K8s resources that were used in the final layout.

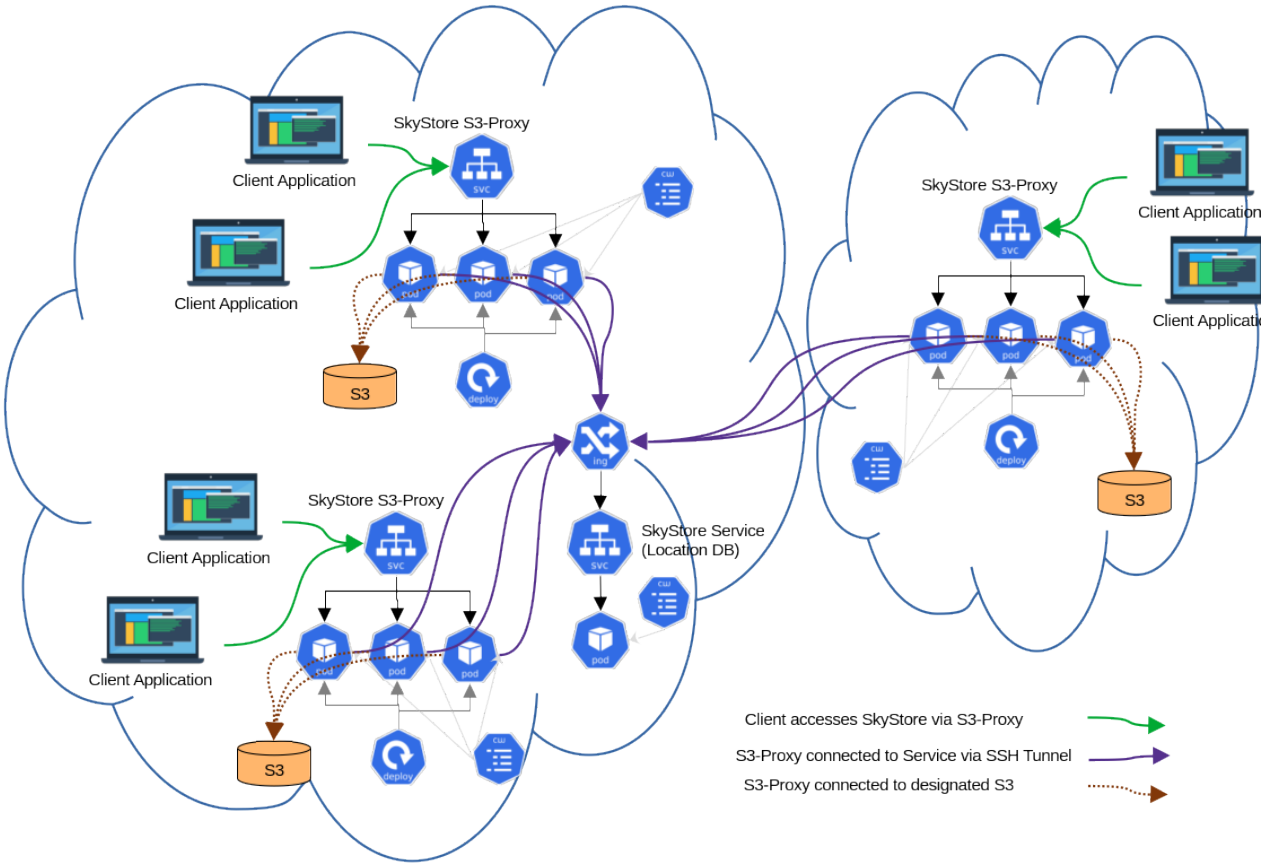

Figure 5. SkyStore on Kubernetes

The first notable difference is the enablement of multi-cluster layout, as shown in Figure 5. This is obtained by configuring the SkyStore Service component as a K8s service [18], which can be made accessible outside the cluster using various well known K8s mechanisms such as Loadbalancer, Ingress, node-port and others [18], depending on what the K8s provider allows. In the case of our EXTRACT tests, we put the SkyStore Service in the OVH cluster, and the allowed exposition method was a Loadbalancer. This resulted in a public IP address and port being assigned to the SkyStore service (in addition to the internal Cluster IP). This allowed all S3 proxies in any other cluster to connect to it.

A second difference of value is the ability to scale S3 proxies that use the same configuration, which means they all connect to the same SkyStore (metadata) Service and to the same local S3. Each S3 proxy is largely stateless, since its function is to delegate operations to different S3 clusters and to the metadata service (for knowledge sharing and bucket management). Hence, it can be safely scaled out and in to mitigate client load. This was obtained by defining an S3 proxy as a service on top of a K8s Deployment [19] (see Figure 5). The deployment gives manual control for scaling the S3 proxy out and in, and the service provides a single entry-point that is load-balanced by K8s over the different replicas.

There are several other smaller K8s-specific changes that should be briefly indicated.

- The configuration of each component is currently a ConfigMap [20], which can be easily derived from the env files of Docker.

- Each S3-proxy, being a service, can also be externalized, similar to the main SkyStore Service, so that it can provide S3 access to clients outside the cluster.

Refinement

Following deployment and integration of SkyStore on K8s with the use-cases with quite a bit of integration tests, several changes have been implemented.

- **Documentation.** Extensive documentation was written for both users and developers as a separate CONTAINERS.md file, and committed to the repository.
- **Failure reflection.** The initial naive versions of the Docker container entrypoints have been improved to fail-fast if the service itself fails, so that the failure is reflected in the container status and detected by Kubernetes for reporting, deployment recovery etc. This required some code changes for SkyStore S3 proxy service, which was originally designed to run separately from its launcher. A different fix was with the SkyStore service, which needs to run both SSH server and the metadata server, and fail when either fails.
- **In-place restart.** Eventually, we added the option to run the SkyStore Service itself as a service on deployment, even though it is stateful (so max replicas is 1). This allows easier in-place restart of the pod in case of software failure without changing or deploying anything else.

3.2.2. S3 Compatibility

A separate issue that was realized with the original version of SkyStore during integration, is that it had no implementation for the S3 API call of *HeadBucket* [21]. This call, which simply checks that a particular bucket exists and is accessible, was left unimplemented. On the other hand, this API call is used by many 3rd-party S3 clients, and specifically that of SIX's Nuvla [22], which implements the data catalog of EXTRACT. SIX identified the issue by testing with their developer version of the Nuvla cloud, and then IBM contributed a fix that was tested and proved to resolve the issue. The fix was pushed to SkyStore repository.

3.2.3. Upstream Merge

Throughout much of the work on EXTRACT, the work on SkyStore in EXTRACT revolved around a repository of https://github.com/gilv/skystore which is actually a fork of the upstream SkyStore repository. This was because EXTRACT work focused on some aspects of SkyStore (discussed above and in D4.2), whereas the upstream owners (UCB) worked with another IBM team on evolving other aspects. When the fork reached a mature state, we began coordinating its merge back into the upstream repository, conjoined with publishing a joint blog about SkyStore with UCB. We (IBM) rebased the fork branch on the latest branch from upstream and submitted it for review. Unfortunately, UCB response was delayed, and by the end of this milestone the merge is not yet complete for delivery. We do hope to have it present for the final project review.

4. Security Enhancement

Up until M30, we have experimented, tested, and validated (in isolation or end-to-end, as applicable) several complementary techniques aimed to bring "secure by design" and "defense in-depth" approaches to securing the overall EXTRACT architecture and its development and deployments. Having these "secure by design" and "defense in-depth" approaches baked into the project from the start, essentially sets the cybersecurity bar high-enough for any further EXTRACT development and uptake, whether by the EXTRACT project members or external stakeholders (e.g., code contributors, commercial licensors, etc.).

Static Analysis

Static analysis of the source code, configurations (e.g., code, testing, environment, deployment), and container/docker images is one of the first and essential steps in having a functional DevSecOps foundation for any modern project and software architecture. Static source code analysis scans the source code for known or very-likely vulnerable source code lines and coding patterns (e.g., unsafe memory management, unsafe system calling, usage of unsafe library functions, etc.). Static analysis of the container/docker images scans the said images for various misconfigurations (e.g., secrets in plain text, insecure and/or default credentials), as well as for known vulnerable third-party libraries/dependencies and sometimes for binary-code vulnerabilities.

Firstly, we have developed our own set of standalone scripts (pluggable to DevOps pipelines of GitLab and GitHub) to (automagically) pull the newly pushed EXTRACT container/docker images and scan them locally for security issues. The local scans are important as this approach gives better and stricter control to the developer and/or end-user whether the container/docker images stay local (hence private), thus avoiding their transfer to some cloud/SaaS service which could expose them to other systems and therefore to other potential external threats (e.g., external leaks, tampering, etc.). With this approach, we employ, among other things, Trivy which is an open-source cybersecurity toolset, thus we stay close to the open-source principles and hope to contribute back any improvements and fixes to Trivy as we discover/develop them.

Secondly, we have complemented our local container/docker static scanning scripts with the state-of-the-art Docker Hub Scout Enterprise scanning [23], which essentially provides similar (and sometimes richer/complementary) security information about the vulnerabilities, weaknesses and suggested remediations within the scanned container/docker images. However, it relies on the cloud/SaaS (usually paid) service from Docker Inc., and while it is optional and suggested, it is not mandatory for the overall functioning of our "Static Analysis" security strategy for EXTRACT.

Thirdly, we have initiated the tested and implementation of security-related static code analysis. For this, we use two approaches. The first approach again is a local scanning performed by a static analysis docker-based tool developed by Binare which integrates and incorporates state of the art static code analyzers (e.g., Qualys [24]). The second approach is a cloud/SaaS static code analysis based on GitHub Enterprise, and while it is optional and suggested, it is not mandatory for the overall functioning of our "Static Analysis" security strategy for EXTRACT.

SBOM Management

Software Bill of Materials (SBOM) [25] becomes an essential part of any software-based system and solution, and will become mandatory especially in the light of EU CRA (Cyber Resilience Act) [26] and similar legal compliance directives. We have gathered SBOM details for each major EXTRACT component through multiple means. First, early in the project, we started with declarative manner, where each owner/developer of the component declares what third-party components, libraries and frameworks they are going to employ, so that we could do an early mapping of the security, vulnerability and CVE [27] surfaces that could potentially affect EXTRACT. In the latter stages, we employed various tools and techniques (including static analysis) to generate the SBOMs in a closer-to-automation manner (while still enforcing human expert oversight to ensure that the SBOMs are accurate and no erroneous components/versions, hence vulnerabilities, are included in the SBOM and vulnerability reports).

As a work-in-progress and for M30-M36 phase, we aim to "close the loop" for SBOM in the sense that we generate the SBOMs automatically, then we import them (e.g., CyclonDX [28], SPDX [29] formats) into Binare Platform (a commercial, proprietary, state-of-the-art platform for cybersecurity), where the imports are done automatically via versatile and flexible APIs exposed by the platform. Subsequently, the imported SBOMs are marked for "continuous SBOM monitoring", which means that the Binare Platform will notify the EXTRACT owner/author of the corresponding SBOM whenever new vulnerabilities become known with cybersecurity and/or exploitation effects on certain software components within the specific SBOM.

Dynamic Analysis and Penetration Testing

Dynamic Analysis and Penetration Testing (including application and network fuzzing) are cybersecurity techniques that complement Static Analysis approaches. While Static Analysis tries to identify vulnerabilities statically, i.e., just "looking" at code/configuration/system without running it, the dynamic analysis (and penetration testing along with fuzzing as subsets of dynamic analysis) try to analyze the system and discover vulnerabilities by actively running the software/system and actively engaging with it with various inputs (both conformant and malformed, as well as benign and malicious inputs).

So far for the purpose of cybersecurity dynamic analysis, we have been focusing on running and executing each software component (e.g., their containers/dockers) in isolation in order to determine the running and execution environment needs and requirements, and also to understand what dynamic analysis and penetration testing tools would best fit for each particular component (e.g., RESTful APIs require some specific tools/approaches, while MQTT [30] protocols require different tools/approaches for penetration testing).

As a work-in-progress and for M30-M36 phase, we aim to "close the loop" from dynamic analysis and penetration testing perspective, in the sense that we aim to have twin EXTRACT deployment of PER and TASKA within Binare's infrastructure with the sole purpose of exposing it to all relevant penetration testing and fuzzing techniques and tools. With this, we aim to essentially simulate what would potential attackers do in the real world should EXTRACT be running in some real-world deployment. We expect the result of this penetration testing should reveal cybersecurity vulnerabilities and weaknesses that we will aim to fix in collaboration with the main owner/author of the impacted component. Moreover, we also aim to add these dynamic analysis and penetration testing steps as automated workflow steps within the bigger DevSepOps [31] vision desired for the EXTRACT project.

Secure Communications

Following secure "data in motion" principles, it is essential to secure the communications between pods/services as well as to/from external world where the end-users and external data comes from. To ensure this at high-level, we have tested vanilla deployments with the following configurations:

- Autocert [32] and step-ca [33] on the ingress controller - this ensures secure and HTTPS-based access and communication from the end-user to the entire cluster/deployment
- mTLS [34] between pods - this ensures secure communication between services within the cluster/deployment

During the Testing, Integration, Evaluation phase of M30-M36, we will apply these techniques and configurations on PER and TASKA evaluation deployments of EXTRACT and will assess their final effectiveness, and if needed will be adjusting/improving the Secure Communication setup and guidelines where blind spots will be detected.

Secrets Management

Properly and safely managing the secrets (e.g., credentials, API tokens, access controls, etc.) is an essential part of any "secure by design" architecture and deployment. Many times, managing this improperly (or lacking such management altogether) leads to insecure and exploitable systems. Generally we address the secrets management as part of the secure "data at rest" principle which is detailed in D2.3. In general, the kubernetes and docker based systems offer tools (e.g., vaults, secure storage) to address the needs of secrets management in a best-practices (and sometimes security-compliant) manner. We will evaluate the effectiveness and limitations of the secrets management (and other security hardening techniques) during the Testing, Integration, Evaluation phase of M30-M36.

General Notes

Last but not least, for the impacted/vulnerable EXTRACT software components, source code, and container/docker images, we have been communicating/reporting on those to the relevant EXTRACT partner and discussing possible ways to remediate and improve the situation on a case-by-case basis. Once again, cybersecurity is not an end-goal which stops, it is a continuous process which requires continuity, because of the evolution/updates of various components and because new vulnerabilities and attacks are discovered on a daily basis and they require attention of the developers/testers.

Moreover, a basic yet comprehensive Vulnerability Disclosure Program (VDP) [35] is in development as work-in-progress. Such a program policy will be tested and validated during the Integration, Testing and Validation tasks within the M30-M36 project period. A VDP is essential (and many times mandatory) for any (online) service(s) and/or software/system, and allows the security bugs and vulnerabilities to be reported in a systematic, responsible, and trackable manner either to the operator of the service and/or to the original author/maintainer of the component impacted by the vulnerability, hence allowing remediation of such vulnerabilities under higher responsibility and accountability standards. Such a template VDP built into EXTRACT should in principle facilitate cybersecurity-compliant uptake and deployment of EXTRACT technology by other use-cases and organizations worldwide.

5. Validation - Demonstrators

5.1. Integrated Demonstrator - WP2/WP4

This demonstrator deals with SkyStore operating as a foundation for the DMF (Data Mining Framework - see D2.3). The scenario is somewhat similar to the cross-premise inference used for the mid-project review. However, it has been extended with the DMF data catalog. Data propagation is done using SkyStore over multiple stores (2 AWS regions), using the Kubernetes version.

The scenario being demonstrated is that a tensor input of inference, after having been generated, is stored on S3 cluster (via SkyStore) and registered in the data catalog. A catalog subscriber gets notified, reads it from a different SkyStore S3-proxy, causing it to propagate to it. Then the tensor input is loaded by the subscriber and used as inference input. See the details in D2.3.

5.2. EXTRACT Deployments Demonstrators

The two videos linked below show the execution of TASKA use case C pipeline using the two public cloud providers tested in section 2.2.

EGI - Rancher : https://b2drop.bsc.es/index.php/s/tGcXHxL7Rt6BHn8

EOSC - OKD : https://b2drop.bsc.es/index.php/s/Teb2fQEdB4mmaL5

6. Conclusion and Future Directions

This deliverable demonstrates successful final release of the EXTRACT continuum layer. The architecture has been implemented and tested in the project use-cases of PER and TASKA. Demonstrators are available showcasing the results.

While the work designated for the project time-frame has completed, there are lots of interesting and valuable future venues to explore building on this foundation, aside from the evaluation of the current product that is scheduled for the next (and final) milestone. We hereby provide a partial list.

SkyStore:

- Auto-scaling S3 proxy
- S3 proxy as a sidecar for clients (alternative to stand-alone service and autoscaling)
- Exploring different replication policies

EXTRACT Security:

- Automating the entire DevOps workflow related to EXTRACT into a complete DevSecOps approach
- Automating SBOM (Software Bill of Material) monitoring and updates (where relevant, applicable, and available) across the variety of involved repositories and technologies used throughout EXTRACT architecture and flows
    - Use EXTRACT's SBOMs inside Binare Platform for monitoring and alerting (the EXTRACT stakeholders) whenever new vulnerabilities are discovered within the third-party components and libraries used throughout the EXTRACT components
- Full security hardening (see also above) of the sensitive "secrets" used throughout various deployment configurations and development pipelines (e.g., using vaults and secure storages layers across all technologies)
- Use the security steps and technologies/tools applied to EXTRACT up-to M36 and develop them into a "blueprint handbook" to be suggested and ideally used as a "Security How-To" in EU funded projects where cybersecurity is not the main focus of the call/project.

7. Acronyms and Abbreviations

- DX.Y – Deliverable X.Y
- K8s – Kubernetes
- TX.Y – Task X.Y
- WP – Work Package
- WPL – Work Package Leader

8. References

[1] "Production-Grade Container Orchestration," Kubernetes. Accessed: Jun. 25, 2025. [Online]. Available: https://kubernetes.io/
[2] "Homepage," EGI. Accessed: Jun. 25, 2025. [Online]. Available: https://www.egi.eu/
[3] "Innovate Everywhere," Rancher Labs. Accessed: Jun. 25, 2025. [Online]. Available: http://www.rancher.com
[4] "Helm." Accessed: Jun. 25, 2025. [Online]. Available: https://helm.sh/
[5] M. Inc, "S3 Compatible Storage for AI," MinIO. Accessed: Jun. 25, 2025. [Online]. Available: https://min.io
[6] "Persistent Volumes," Kubernetes. Accessed: Jun. 25, 2025. [Online]. Available: https://kubernetes.io/docs/concepts/storage/persistent-volumes/
[7] "Ceph.io — Home." Accessed: Jun. 25, 2025. [Online]. Available: https://ceph.io/en/
[8] "Lithops." Accessed: Jun. 25, 2025. [Online]. Available: https://lithops-cloud.github.io/
[9] "EOSC Association," EOSC Association. Accessed: Jun. 25, 2025. [Online]. Available: https://eosc.eu/
[10] "Kubernetes at Scale on any Infrastructure | OKD Kubernetes Platform." Accessed: Jun. 25, 2025. [Online]. Available: https://okd.io/
[11] "Red Hat OpenShift enterprise application platform." Accessed: Jun. 25, 2025. [Online]. Available: https://www.redhat.com/en/technologies/cloud-computing/openshift
[12] "eduGAIN – enabling worldwide access." Accessed: Jun. 25, 2025. [Online]. Available: https://edugain.org/
[13] "Amazon Simple Storage Service Documentation." Accessed: Jun. 25, 2025. [Online]. Available: https://docs.aws.amazon.com/s3/
[14] "Ceph File System — Ceph Documentation." Accessed: Jun. 25, 2025. [Online]. Available: https://docs.ceph.com/en/reef/cephfs/
[15] S. Liu *et al.*, "SkyStore: Cost-Optimized Object Storage Across Regions and Clouds," Feb. 28, 2025, *arXiv*: arXiv:2502.20818. doi: 10.48550/arXiv.2502.20818.
[16] "What is a Container? | Docker." Accessed: Jun. 25, 2025. [Online]. Available: https://www.docker.com/resources/what-container/
[17] "Swarm mode," Docker Documentation. Accessed: Jun. 25, 2025. [Online]. Available: https://docs.docker.com/engine/swarm/
[18] "Service," Kubernetes. Accessed: Jun. 25, 2025. [Online]. Available: https://kubernetes.io/docs/concepts/services-networking/service/
[19] "Deployments," Kubernetes. Accessed: Jun. 25, 2025. [Online]. Available: https://kubernetes.io/docs/concepts/workloads/controllers/deployment/

[20]"ConfigMaps," Kubernetes. Accessed: Jun. 25, 2025. [Online]. Available: https://kubernetes.io/docs/concepts/configuration/configmap/
[21]"HeadBucket - Amazon Simple Storage Service." Accessed: Jun. 25, 2025. [Online]. Available: https://docs.aws.amazon.com/AmazonS3/latest/API/API_HeadBucket.html
[22]"Nuvla." Accessed: Jun. 25, 2025. [Online]. Available: https://nuvla.io/ui/
[23]"Software Supply Chain Management for Developers | Docker Scout." Accessed: Jun. 25, 2025. [Online]. Available: https://www.docker.com/products/docker-scout/
[24]"Vulnerability and Web Application Scanning Accuracy | Qualys." Accessed: Jun. 25, 2025. [Online]. Available: https://www.qualys.com/scanning-accuracy/
[25]"Software Bill of Materials (SBOM) | CISA." Accessed: Jun. 25, 2025. [Online]. Available: https://www.cisa.gov/sbom
[26]"Cyber Resilience Act | Shaping Europe's digital future." Accessed: Jun. 25, 2025. [Online]. Available: https://digital-strategy.ec.europa.eu/en/policies/cyber-resilience-act
[27]"CVE: Common Vulnerabilities and Exposures." Accessed: Jun. 25, 2025. [Online]. Available: https://www.cve.org/
[28]"CycloneDX Bill of Materials Standard | CycloneDX." Accessed: Jun. 25, 2025. [Online]. Available: https://cyclonedx.org/
[29]"Security – SPDX." Accessed: Jun. 25, 2025. [Online]. Available: https://spdx.dev/learn/areas-of-interest/security/
[30]"MQTT - The Standard for IoT Messaging." Accessed: Jun. 25, 2025. [Online]. Available: https://mqtt.org/
[31]Home *et al.*, "DevSecOps," DevSecOps. Accessed: Jun. 25, 2025. [Online]. Available: https://www.devsecops.org
[32]M. Martini, "Autocert Introduction and Tutorial," Kubecademy. Accessed: Jun. 26, 2025. [Online]. Available: https://cozykube.com/autocert/
[33]"`step-ca` server." Accessed: Jun. 26, 2025. [Online]. Available: https://smallstep.com/docs/step-ca/
[34]"What is mTLS? | Mutual TLS." Accessed: Jun. 26, 2025. [Online]. Available: https://www.cloudflare.com/learning/access-management/what-is-mutual-tls/
[35]"Vulnerability Disclosure Program (VDP)," Bugcrowd. Accessed: Jun. 26, 2025. [Online]. Available: http://www.bugcrowd.com/glossary/vulnerability-disclosure-program-vdp/