ESCAPE prototypes a data infrastructure for open science

Rosie Bolton[2], *Simone* Campana[2,*], *Andrea* Ceccanti[3], *Xavier* Espinal[2] , *Aristeidis* Fkiaras[2] *Patrick* Fuhrmann[4] and *Yan* Grange[5]

[1]SKAO, Jodrell Bank, Cheshire, SK11 9FT, UK
[2]CERN, 1 Esplanade des Particules, 1211, Meyrin, CH
[3]INFN CNAF, Viale Carlo Berti Pichat 6, 40127 Bologna, Italy
[4]DESY, Notkestrasse 85, 22607 Hamburg, DE
[5]ASTRON, Oude Hoogeveensedijk 4, 7991 PD Dwingeloo, NL

Abstract. The European-funded ESCAPE project will prototype a shared solution to computing challenges in the context of the European Open Science Cloud. It targets Astronomy and Particle Physics facilities and research infrastructures and focuses on developing solutions for handling Exabyte scale datasets. The DIOS work package aims at delivering a Data Infrastructure for Open Science. Such an infrastructure would be a non HEP specific implementation of the data lake concept elaborated in the HSF Community White Paper and endorsed in the WLCG Strategy Document for HL-LHC. The science projects in ESCAPE are in different phases of evolution. While HL-LHC can leverage 15 years of experience of distributed computing in WLCG, other sciences are building now their computing models. This contribution describes the architecture of a shared ecosystem of services fulfilling the needs in terms of data organisation, management and access for the ESCAPE community. The backbone of such a data lake will consist of several storage services operated by the partner institutes and connected through reliable networks. Data management and organisation will be orchestrated through Rucio. A layer of caching and latency hiding services, supporting various access protocols will serve the data to heterogeneous facilities, from conventional Grid sites to HPC centres and Cloud providers. The authentication and authorisation system will be based on tokens. For the success of the project, DIOS will integrate open source solutions which demonstrated reliability and scalability as at the multi petabyte scale. Such services will be configured, deployed and complemented to cover the use cases of the ESCAPE sciences which will be further developed during the project.

* Corresponding author: Simone.Campana@cern.ch

1 Introduction

The European Science Cluster of Astronomy and Particle physics ESFRI research infrastructure – ESCAPE [1] – is a European Union funded project in the context of Horizon 2020 [2]. The cluster is formed by science projects with Exabyte-scale computing and storage needs in the 2020s and the main goal of ESCAPE is prototyping a digital infrastructure for those needs. Ultimately, ESCAPE should ensure that the sciences it represents drive the development and evolution of the European Open Science Cloud – EOSC [3]. The goal of the ESCAPE Work Package 2 (WP2 DIOS - Data Infrastructure for Open Science) is to build a cloud of data services, often referred as datalake. The datalake should serve as core infrastructure to support open data and enable the FAIR principles, by providing a flexible and scalable infrastructure to store and access scientific data, while optimizing the total cost of ownership.

2 Datalake architecture

The architecture of the datalake in terms of functional elements is described in Fig.1. The core of the infrastructure consists of the storage services at the different facilities. These facilities differ in size and expertise and will be able to provide different classes of service: large national centres operate archive storage (today based on tape media) as well as smaller disk based systems to serve compute intensive data processing. Other facilities operate disk based solutions only, and in some cases such storage is volatile. The storage can be deployed as a distributed service, spanning multiple physical facilities but offering a single entry point for the users. The storage services rely on different software technologies, which need to interoperate by following the defined interfaces for file access, file transfer and storage management. Data is replicated across the different storage services asynchronously through transfer scheduling service. The data centres constituting the backbone of the datalake infrastructure are connected by fast networks, i.e. with at least multiple 10Gb/s links. A higher level service organizes the storage and orchestrates the data inside the datalake. Such service offers functionalities for cataloguing the data files with

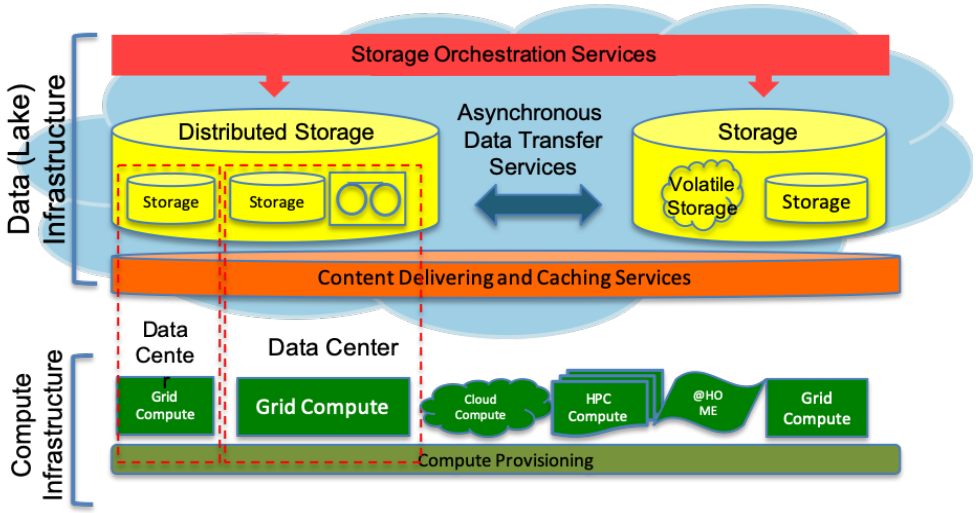

Fig. 1. Diagram showing the building blocks of the datalake infrastructure and the connection with compute services.

some metadata, organizes the files in higher level structures such as datasets, and stores file locations in the datalake. It also allows the definition of policies for data replication and data deletion, and enforces them through a set of asynchronous agents. The resources to compute the data in this model could be co-located with the data, i.e. at the same data centre where the data are stored, but in general one should expect that not to be the case. A content delivery service is needed to hide the effects due to network latency in data access. At the same time, the service should be able to cache the most-accessed data close to the computing resources, to optimize data access performance in case of re-use. The content delivery and caching layer should be able to serve data to a heterogeneous set of resources, ranging from sites accessible through Grid interfaces, to Cloud resources, to High Performance Computers. The Computing Provisioning layer, orchestrating workflows and pipelines is also shown in Fig. 1, while not part of the WP2 scope. Finally, given the heterogeneous and distributed nature of the system, the Authentication, Authorization and Identity management (AAI) play a crucial role in the architecture. The system must accommodate the needs of the ESFRIs in terms of policies and allow open access outside the data embargo periods. At the same time, the mechanisms for authentication and authorization need to scale for exa-scale data management, reducing the overhead, while enforcing an adequate level of security.

3 Datalake components and reference implementation

Different components that will be used to build the datalake reference implementation are described in the following. These will be also accessible to the ESFRIs as stand-alone services. Some elements will necessitate R&D work to reach the level required within ESCAPE. This activity will be part of the WP2 implementation plan.

3. 1 Storage Technologies

We plan to build the datalake leveraging a heterogeneous set of storage solutions: dCache [4], DPM [5], EOS [6], StoRM [7] and xrootd [8] at the minimum. Such technologies have been deployed for many years in the Worldwide LHC Computing Grid (WLCG) [9] infrastructure and demonstrated their capability to operate at the hundred petabyte scale. Some storage solutions such as dCache, EOS and DPM are used also in federating geographically distributed data centres under a unique storage namespace. Such solution will also be included in the ESCAPE datalake prototype. The data access will be enabled through a set of protocols. The xrootd protocol will be supported particularly for the High Energy Physics (HEP) use case, where files are stored in a ROOT [10] format. The HTTP protocol will be supported to cover the use case of most ESCAPE science projects as it is the most widely adopted within the open source community. Both xrootd and HTTP offer functionalities to upload/download and stream files, trigger third party transfers (more on this in the File Transfer Service Section), perform basic storage management operations and delete files. Storages should be accessible through x509 credentials, but should also support token based authentication, which will be the reference solution in ESCAPE, see section 2.5. For the future optimization of storage cost and performance, Quality of Service (QoS) will be a key element. So far disk and tape have been used as storage classes to implement online and archive storage, respectively, but in ESCAPE we intend to consider a richer set of QoS metrics and a corresponding set of capabilities at the storage level. We are focusing on three basic metrics: Reliability, Performance and Cost, and we intend to enable access to storage resources corresponding to combinations of such capabilities.

3. 2 Asynchronous Data Transfer Services and Networking

The WLCG File Transfer Service - FTS [11] has been used in production for LHC experiments for more than a decade. It went through a major refactoring a few years ago to take advantage of modern technologies and improve scalability and performance. It has demonstrated the capability of efficiently performing file replication in a distributed infrastructure at the level of many millions of files per day, consisting of multiple petabytes. In ESCAPE we plan to use FTS as reference implementation for an asynchronous data transfer service across sites of the datalake. However, we expect R&D work in this area to meet the ESCAPE goals. The plan is to replace the gridFTP protocol, in use for many years for third party copy in WLCG, with alternatives better supported as open source software. Two viable alternatives have been identified in the xrootd and HTTP protocols. In ESCAPE we plan to complement the already ongoing effort in commissioning those protocols for third party copy, ensuring they are well supported by the various storage solutions and integrated in FTS. The commissioning should be seen both at the level of functionality, including verification of file integrity and metadata handling, and performance, through a set of stress tests and data challenges. We plan also to use FTS as an engine to trigger data staging from tape to disk buffers in an organized way. Its capability in this respect has been already demonstrated by the LHC experiments. We plan to use the FTS monitoring and the capabilities of its data analytics backend for most of the commissioning activity. We will also expose the monitoring data through a set of APIs for the different ESCAPE ESFRIs to consume and possibly build specific dashboards. At the same time, we will instrument the infrastructure with a network level monitoring system. For this, we agreed to use the perfSONAR [12] technology as it is already deployed at most of the data centres participating in the ESCAPE datalake. The plan is to ensure that each storage site is instrumented with a perfSONAR instance close, in network proximity, to the storage service. We will schedule a set of regular tests through perfSONAR, probing packet loss and throughput between all sites. The results will be displayed in an ESCAPE perfSONAR dashboard and accessible again through an API. We finally intend to participate to the NOTED [13] R&D effort, setting up a mechanism to tailor network paths based on the load as measured by high level services such as FTS.

3. 3 Storage Orchestration Service

The orchestration service plays a central role in the architecture of the datalake, as explained in the previous section. We agreed to use Rucio [14] as a reference implementation of the orchestration service in ESCAPE. Rucio is an open-source data management system for scientific computing. It has been initially developed in the scope of the ATLAS [15] experiment and adopted by other HEP experiments such as CMS [16] and DUNE [17]. It was positively evaluated by many ESCAPE science projects. Rucio is able to manage data at a file granularity level and to organize them in collections such as archives, datasets and containers. “File” and “collection” definitions can be enriched with some user-defined metadata. The physical management of data in Rucio is driven by “rules”. A rule is a user-defined policy declaring the expectations in terms of data replication for a particular file or dataset: number of replicas and geolocation are examples of rule attributes. Rucio leverages FTS for file transfers and the GFAL [18] library for file deletion. Both replication and deletion rely on the gridFTP/xrootd/HTTP protocols, consistent therefore with the plans described in the previous two sections. In addition, Rucio offers an extensive set of monitoring tools. Information are collected in a BigData infrastructure and can be used to generate customizable dashboards as well as to perform detailed performance studies and operational debugging. In ESCAPE, we intend to rely on

the CERN Agile Infrastructure [19] to store, organize and expose such monitoring data. While Rucio offers the core functionalities necessary for an ESCAPE orchestration service, R&D is required, as well as the commissioning and improvement of several components, to meet the needs of different science projects. Rucio allows storing and retrieving some metadata, whereas the full support for user-defined metadata is at a prototype stage. In ESCAPE, we intend to evaluate the current prototype and improve it to accommodate the use cases of the ESCAPE ESFRIs, or propose an alternative implementation if needed. The concept of QoS in storage needs to be integrated into the orchestration service. Therefore, Rucio needs to evolve to acquire the knowledge of QoS classes and consider them as part of the storage system attributes when defining new storage endpoints. Special attention needs to be paid in transitions between classes as Rucio will need to handle them. Finally, Rucio needs to support the transition between X509 authentication and token based authentication as will be explained in more details in section 2.5.

3. 4 Content Delivery and Caching

In the datalake model, the processing resources might not be necessarily co-located with the data. Different models can be foreseen to deliver data content to processing units. In fact, from the discussion with the ESCAPE science projects it was clear that we need to support two scenarios, as detailed in the following. In the first scenario, the application running in a batch cluster or interactively needs to access remote data. The protocol connects to a content delivery layer, which redirects to a storage location containing the file. The content delivery layer fetches the data and streams it to the client. The ability of the content delivery layer to buffer data allows the client to process the data as if it was local, provided that the application is CPU bound and not I/O bound. The latency due to the network distance will be perceived by the client only at the beginning of the process, when the first part of the data needs to be buffered from the storage to the content delivery. Afterwards, the latency is not perceived by the application. In the second scenario, the application running in a batch cluster or interactively tries to download data from the content delivery layer. Such system, enabled with some storage capacity, serves the data directly to the client. In case the data is not present, the content delivery layer will fetch it from the storage location containing the file, cache it locally, and serve it to the client. The content of the cache will be kept for a period of time depending on the cache size and the number of requests. The cache system will delete the data depending on a set of conditions, for example the least recently used data first. We identified the xCache technology [20] as the most promising option fulfilling the requirements of the ESCAPE. We will therefore adopt it as reference implementation for the content delivery layer in the datalake. Some studies and R&D will be needed in this area. Both scenarios should be tested with real applications from the ESCAPE science projects.

3. 5 Authentication, Authorization and Identity services

A common and flexible Authentication and Authorization Infrastructure (AAI) is a key requirement to enable secure and controlled data access in the datalake. The ESCAPE project will not invent new authentication and authorization mechanisms but will build upon existing work, leveraging the 15-years experience of WLCG in building a global AAI and the recent results of the INDIGO-Datacloud [21] and AARC [22] projects.
The INDIGO Identity and Access Management (IAM) [21] service will be the central enabling technology for the ESCAPE AAI. IAM is an integrated identity solution developed at INFN as the evolution of VOMS [22] and has been selected as the building block for the next-generation AAI for WLCG. In this next-generation AAI, services expose

functionality through OAuth [23] protected APIs: only agents presenting a valid and trusted access token are granted access. Access tokens, which are signed Json Web Tokens (JWTs), can be obtained by client applications (browsers, command line interfaces or other services) from the central IAM service. Access tokens, depending on the IAM configuration and community requirements, can provide identity information (e.g., an opaque user identifier, groups, and other attributes) and other authorization information (e.g., capabilities). Authorization is then performed at services based on the token contents, after a token verification step that assesses token integrity and validity. Some services may require the exchange of the science project issued access token with a local, service-specific one issued by the service itself that is then used to drive authorization decisions. In both scenarios, the agent is authorized based on the information asserted by the central IAM service in the initial access token. Legacy services are integrated via token translation, ie. without requiring changes to their codebase and authentication/authorization logic.

3. 6 Information and configuration system

The ESCAPE datalake integrates many distributed services fulfilling different roles and those services need to interoperate. The topology of the services and the information about how to access their interfaces needs to be stored in an information system and exposed through a REST API. For this functional element we agreed to use the CRIC [25] technology as reference implementation. CRIC is a central catalogue containing services information. A core module aggregates generic information about storage and processing services. This information can be complemented and decorated with community specific configuration through a set of plugins.

3. 7 Monitoring

Logs of the services running on the infrastructure will be collected, stored and visualized using industry standard solutions like the Elastic Stack [26]. Events being produced by the Rucio Server will also be stored on the same solution. For Third Party Transfers in particular we will also use existing FTS monitoring dashboards to enrich our view. The caching layer will be integrated with dedicated monitoring relying on different metrics. The combination of those solutions will allow us to have a real-time view of the performance and load of the datalake as well as spot and investigate potential issues or downtime.

4. Datalake integration and deployment

The WP2 implementation workplan consists of two phases, as defined in the ESCAPE proposal. The pilot phase, lasting 18 months will focus on demonstrating the data lake model, by providing a small scale but functional system, integrating the technologies identified in the preparation phase. The prototype phase will focus on deploying a full scale system, allowing functional tests as well as stress tests of all the capabilities needed by the ESCAPE science projects for FAIR data management.

5. Conclusions

In the first 8 months of the project we went through the process of collecting requirements from the involved science projects. We also went through a series of technical meetings highlighting the available technologies that could be candidates building blocks of the ESCAPE datalake. We defined an architecture of such system and elaborated an

implementation plan that would deliver in steps of increasing complexity all functionalities needed by our science communities. The milestones for this plan would be the ones described in the ESCAPE proposal document, together with the next deliverables at the end of the pilot and prototype phase.

References

1. https://projectescape.eu [accessed 2020-06-05]
2. https://ec.europa.eu/programmes/horizon2020/en [accessed 2020-06-05]
3. https://www.eoscsecretariat.eu [accessed 2020-06-05]
4. http://dcache.org [accessed 2020-06-05]
5. http://lcgdm.web.cern.ch/dpm [accessed 2020-06-05]
6. https://eos.web.cern.ch/ [accessed 2020-06-05]
7. https://italiangrid.github.io/storm/index.html [accessed 2020-06-05]
8. http://xrootd.org [accessed 2020-06-05]
9. http://wlcg.web.cern.ch/ [accessed 2020-06-05]
10. https://root.cern.ch/ [accessed 2020-06-05]
11. https://fts.web.cern.ch/ [accessed 2020-06-05]
12. https://www.perfsonar.net/ [accessed 2020-06-05]
13. http://cern.ch/go/lkr7 [accessed 2020-06-05]
14. https://rucio.cern.ch/ [accessed 2020-06-05]
15. http://atlas.cern [accessed 2020-06-05]
16. http://cms.cern [accessed 2020-06-05]
17. https://www.dunescience.org [accessed 2020-06-05]
18. https://dmc.web.cern.ch/projects/gfal-2/home [accessed 2020-06-05]
19. https://doi.org/10.1051/epjconf/201921408031 [accessed 2020-06-05]
20. https://www.indigo-datacloud.eu [accessed 2020-06-05]
21. https://aarc-project.eu [accessed 2020-06-05]
22. https://italiangrid.github.io/voms/ [accessed 2020-06-05]
23. https://oauth.net/2/ [accessed 2020-06-05]