# *HPC integration handbook*

| | |
|---|---|
| **Lead partner:** | EGI Foundation |
| **Version:** | 1.0 |
| **Status:** | Final |
| **Dissemination Level:** | Public |
| **Keywords:** | HPC, EOSC, Compute, Platform |
| **Document Link:** | https://documents.egi.eu/public/ShowDocument?docid=3875 |


**Abstract**

The first version of the HPC integration handbook describes the EGI-ACE activities to extend the EOSC Compute Platform with High Performance Compute (HPC) systems. The work covers integration of HPC systems with the federation tools of the Compute Platform, and the validation of the setup through scientific workflows that use Cloud, High Throughput Compute (HTC) and HPC resources in a mixed way. The handbook details the integration mechanisms used by the HPC systems and provides information on how to use them effectively for running container-based workloads.
This document is the first version of the handbook at half-time of the integration work. A complete version will be reached in June 2022 and a final version of this handbook will be published then as Deliverable 7.3.



EGI-ACE receives funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no. 101017567.


go.egi.eu/egi-ace

## COPYRIGHT NOTICE



EGI-ACE receives funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no. 101017567.

## DELIVERY SLIP

| Date | Name | Partner/Activity |
|---|---|---|
| **From:** | Enol Fernández | EGI Foundation / WP7 |

## DOCUMENT LOG

| Issue | Date | Comment | Author |
|---|---|---|---|
| **v.1** | 31/01/2022 | First version of the handbook | F. Antonio (CMCC),<br>H. Bayındır (TUBITAK),<br>I. Díaz (CESGA),<br>M. Dulea (IFIN-HH),<br>C. Fernández (CESGA),<br>E. Fernández (EGI Foundation),<br>J. Gomes (LIP),<br>A. Lahiff (UKAEA),<br>D. Southwick (CERN),<br>D. Spiga (INFN),<br>G. Sipos (EGI Foundation) |

## TERMINOLOGY

https://confluence.egi.eu/display/EGIG

**Contents**


Executive summary ..... 4

1 Introduction ..... 5

1.1 Heading 2 ..... **Error! Bookmark not defined.**

1.1.1 Heading 3 ..... **Error! Bookmark not defined.**

2 Another example of heading 1 ..... **Error! Bookmark not defined.**

2.1 Heading 2 ..... 6

2.1.1 Heading 3 ..... **Error! Bookmark not defined.**

2.1.1.1 Heading 4 ..... **Error! Bookmark not defined.**

2.1.1.1.1 Heading 5 ..... **Error! Bookmark not defined.**

2.1.1.1.1.1 Heading 6 ..... **Error! Bookmark not defined.**

3 Now for the lists etc. ..... **Error! Bookmark not defined.**

3.1 Bullet lists ..... **Error! Bookmark not defined.**

3.2 Numbered lists ..... **Error! Bookmark not defined.**

4 Figures and captions ..... **Error! Bookmark not defined.**

4.1 Pictures and text ..... **Error! Bookmark not defined.**

4.2 Tables ..... **Error! Bookmark not defined.**

5 References ..... **Error! Bookmark not defined.**

# Executive summary

The EOSC Compute Platform, delivered by the EGI-ACE project, is a system of federated compute and storage facilities, complemented by diverse access, data management and compute platform services. The EOSC Compute Platform is designed to support a wide range of scientific data processing and analysis use cases, including the hosting of scientific services and data spaces. The infrastructure layer of the EOSC Compute platform initially builds on compute cloud, container and High Throughput Compute facilities. During EGI-ACE this layer is extended with HPC systems, moreover the access to the federated HPC resources will be demonstrated by Open Science workflows that span across all EOSC Compute platform continuum. The HPC integration work in EGI-ACE is focused on the following technical areas:

1. EOSC-compliant federated user access management on HPC systems.
2. Availability and reliability monitoring of federated HPC sites.
3. Integrated usage accounting across HPC, cloud and HTC sites.
4. Access to distributed, federated data from HPC systems.
5. Portable container-based applications for cloud compute, HTC and HPC systems.

These areas are investigated by 4 HPC centres, CESGA (Spain), IICT-BAS (Bulgaria), LIP/INCD (Portugal), and TUBITAK (Turkey), all members of the EuroCC project. The findings will be configured on these sites and will be validated through 4 scientific workflows:

- High Energy Physics (HEP) simulations for the High Luminosity run of the Large Hadron Collider (CERN)
- PROMINENCE - cross platform fusion workflows (UK Atomic Energy Authority)
- Photon and neutron science use case from the ELI-NP Research Infrastructure (IFIN-HH, Romania)
- Climate research use case from ENES (CMCC, Italy)

The work will deliver an architecture blueprint for HPC providers about how to provision HPC resources with federated access management, monitoring, accounting, data and application access mechanisms in EOSC. The blueprint will be a document deliverable in June 2022. EGI-ACE will engage with external HPC providers after that to boost the uptake, and ultimately increase the presence and accessibility of HPC systems via the EOSC Compute Platform, and the EOSC Portal. This document is produced at half-way towards this blueprint.

# 1 Introduction

The EOSC Compute Platform, delivered by the EGI-ACE project, is a system of federated hybrid compute and storage facilities, research data hosting, processing and analytics tools, and a set of complementary services for distributed data and compute access to support processing and analytics for distributed data and compute use cases. The Platform builds on the existing EGI infrastructure and seeks to expand it towards HPC systems to support heterogeneous computing workflows that combine the use of HTC, HPC and Cloud resources.

By June 2022, under the guidance of four international use cases, the project will deliver an architecture blueprint for the provisioning of HPC resources in EOSC. The blueprint defines integrational paths for federated access management, monitoring, and usage accounting, solutions that promise to lower the barrier of access to HPC systems that are traditionally not exposed to international access mechanisms.

Our work is focused on four areas: (1) access to HPC systems via policies, solutions and protocols supporting federated identities; (2) the external publication of HPC usage metrics; (3) the execution of portable container-based applications across IaaS, HTC and HPC systems, and (4) federated data access from HPC systems.

This handbook is delivered by the HPC integration task members of EGI-ACE that include:

- 4 HPC providers belonging to National Competence Centres (NCCs) of the EuroCC project (H2020 project with 50% funding from EuroHPC JU, 50% national funding). NCCs are the central points of contact for HPC and related technologies in their country. EGI-ACE includes as providers: CESGA (ES), IICT-BAS (BG), LIP/INCD (PT), and TUBITAK (TR)
- 4 application pilots with applications from different disciplines: climate research with a use case from ENES; High Energy Physics with a use case looking into the computational needs of the future High Luminosity run of the Large Hadron Collider (HL-LHC); fusion with a platform to run workloads across different computing infrastructures , and photon and neutron science with a use case from ELI-NP.
- 1 technology provider (INFN) with broad experience in delivering solutions for HTC, HPC and cloud computing.

This first version of the handbook is organised as follows: First, we will describe the requirements from the pilots included in the HPC integration task. Then, the document details the different options explored for providing access to the available systems and to make them an integral component of the EOSC Compute Platform. The section after those details how users can execute their workflows as containers in the HPC systems. Last, conclusions are given.

# 2 Requirements from pilots

## 2.1 HEP

The High Luminosity run of the Large Hadron Collider (HL-LHC) in 2027 is expected to produce 1 Exabyte of physics data for processing, of which, the goal is to demonstrate processing of 1 Petabyte of physics data in 24 hours through an HPC site. In preparation of this goal, HEP pilot work has identified requirements in line with the thematic topics presented in this milestone document.

Due to the large volume of data that must be transferred to and from a participating HPC site, as well as potentially stored on shared file systems, it is critical that detailed information covering site connectivity and storage capabilities is published alongside applicable usage policies governing these services. Physical descriptions of the network and storage topologies alongside relevant benchmarks demonstrating expected throughput capabilities greatly aids data-driven workload users of HPC sites. Support of data transfer services (XrootD, globus, gftp, etc.), their protocols, and expected throughputs, allows better informed decisions on workload selection and scaling, and reduces potentially invasive benchmarking.

Currently, HEP workloads are served from CVMFS when available or containerized via Singularity images. The lack of a clear consensus on a preferred containerization service for HPC requires communication from site operators on what service(s) they choose to support, and any affecting permission policies enforced. If compute/worker nodes are restricted to LAN connectivity, this should be clearly communicated, as this entails intermediate storage on the shared storage system as an additional step to export/publish results. HEP workloads are diverse and will be heterogeneous in nature. AAI is detailed in the following sections.

## 2.2 PROMINENCE

The PROMINENCE platform allows users to transparently run containerised workloads, including individual jobs as well as workflows, across any number of clouds simultaneously. Users are presented with a simple batch-system style interface available as either a CLI which can be run anywhere or a REST API. All infrastructure provisioning is handled automatically and is completely invisible to users.

Integrating PROMINENCE with existing HPC systems would allow users to access a wider variety of resources and enable users to run multi-node jobs requiring low-latency interconnects more readily, as such hardware is still rarely available in private research clouds. Requirements for HPC integration are:

- A minimum of CentOS 7 (or equivalent).
- Support for unprivileged containers, either Singularity or udocker.
- Ideally outgoing access to the internet on port 443 (https) is required for access to external storage systems. If outgoing access is only available on login nodes this can be dealt with but access from the worker nodes is preferred.
- Access via ssh is fine but HTC-like access, e.g. ARC CE or HTCondor CE, could also be easily supported if available.

- The HPC site's security policies need to allow a single system to be able to submit jobs on behalf of multiple users. PROMINENCE users are of course not given direct access to the resources but are able to submit jobs to PROMINENCE using arbitrary container images running arbitrary commands.

## 2.3 ELI-NP

The 10 PetaWatts High Power Laser System (HPLS) - commissioned in 2020, and the Variable Energy Gamma-ray (VEGA) System - under construction, hosted at the Extreme Light Infrastructure – Nuclear Physics (ELI-NP) facility, will support breakthrough experiments in laser and nuclear physics whose preparation, optimisation and validation need intensive HPC simulations. The largest consumers of HPC resources are the particle-in-cell (PIC) simulations, that are essential for predicting the results of the experimental investigation on ion acceleration and QED effects, laser-to-gamma conversion, for the development of nuclear detector systems, etc.

ELI-NP pilot is intended to meet in general the HPC user requirements and in particular the necessity for providing them with reliable PIC computing resources. User access to open-source codes such as EPOCH and PIConGPU can be offered on bare-metal clusters through the AAI described in this document, or on virtual clusters – through the EGI Check-in service. Generally, the virtual HPC clusters are served through pre-defined, containerized VM images. Efficient workload managers like SLURM should be provided on virtual clusters too.

In order to ensure superior scalability, the bandwidth available for internal communication should be at least 50 Gb/s. On high density clusters Mellanox EDR should be considered. For fast external data transfers it is recommended to have a connectivity of min. 10 Gb/s between providers and users.

The computational power and the storage capacity required depend strongly on the nature of the simulated experiment and of the spatio-temporal scale chosen for the simulation. As a typical example, the running of a 2-dimensional PIC code for investigating gamma-ray generation and pair production typically requires at least 600 CPU cores. To run a 3-dimensional code, more than 1000 cores are needed, and the generated data (including the restart files) are of the order of 1 TB. In general, the time required for running the codes decreases with the number of cores used.

The preparation by the provider of the resources for a new PIC computing project requires close communication with the user for serving appropriate computing environments and preliminary benchmarks of the chosen software on available resources.

## 2.4 ENES

In several domains, such as climate science, scientific advances now rely on technologies and software solutions from both the HPC and Big Data landscapes. However, being able to efficiently exploit HPC infrastructures for running scientific data analysis is not trivial. A unified model that also allows the deployment on HPC of the same services already

exploited in the cloud can pave the way for a wider range of opportunities in the scientific community, further fuelling the adoption of the HPC as a Service (HPCaaS) paradigm. In this respect, software containers are good candidates for supporting portability and deployment of data analytics frameworks over multiple platforms. Thanks to the recent development of HPC-friendly container technologies (e.g., udocker, Singularity, Sarus), scientists could exploit the benefits of this model also on HPC infrastructures.

The ENES pilot operates on top of the ENES Climate Analytics Service (ECAS)[1], one of the EOSC-Hub Thematic Services as well as a Compute Service in the IS-ENES3 project. In the European Open Science Cloud (EOSC) context, ECAS represents a central component of the ENES Data Space[2] set up in the EGI-ACE project with the aim to provide an open and cloud-enabled data science environment for climate scientists. In this environment, the Ophidia HPDA framework[3] represents the core computing engine of the ECAS service and it can greatly benefit from the exploitation of HPC resources for running data analysis applications and workflows.

In such a context, the main goal of the ENES pilot is to explore solutions for the execution of container-based climate workloads, focusing on simplifying the portability of applications across the different computing services available to EOSC users. Specifically, in order to enable a transparent and portable deployment of ECAS on top of the HPC resources made available in the EGI infrastructure, the pilot targets the use of unprivileged containerization solutions (i.e., udocker) for the ECAS core components. Among them, the Ophidia platform, a JupyterHub instance as an entry point to a data science environment and other climate community tools and Python Data Science modules. This will allow climate scientists to easily execute docker-based data analytics and visualization jobs on scientific datasets hosted on the EGI DataHub[4]. To this end, the proper orchestration of container-based jobs for climate applications via HPC batch scheduler (e.g., Slurm) will also play an important role.

The ENES pilot comprises two main scenarios. The first one consists of a single image including a set of climate community-based tools deployed on HPC for sequential and parallel analysis. It also aims at showcasing a first integration with other services like AAI (e.g., EGI Check-in) and federated data access (e.g., EGI DataHub). A second, more complex scenario will target distributed, larger-scale analysis. In this case, a server container will host several components such as the Jupyter notebook server, the Ophidia centralized components and Python modules for result post-processing and visualisation; additionally, multiple worker containers will be deployed on multiple cluster nodes, to enable larger scale parallel data analysis.

---

[1] https://marketplace.eosc-portal.eu/services/enes-climate-analytics-service
[2] https://enesdataspace.vm.fedcloud.eu
[3] https://ophidia.cmcc.it
[4] https://www.egi.eu/services/datahub/

# 3 Integration of HPC Systems

## 3.1 Access - Authentication and Authorization

Traditionally, access to HPC systems is performed via login nodes where users connect to with SSH. From those nodes, users can interact with the batch system and submit jobs for their execution. The SSH credentials are typically locally managed usernames and password or SSH keys. For EGI-ACE, we are testing new ways of accessing the providers that leverage federated authentication, so users do not need to manage a new set of credentials for each of the individual systems. The following mechanisms can be considered for integration:

- SSH access with OIDC credentials. This enables access to SSH using the OIDC federated identity technology as supported by EGI Check-in. This is the preferred authentication mechanism for the EGI-ACE pilots as it minimises the requirements to the users, who just need to get access tokens from Check-in. SSH access may be also enabled using SSH keys from a federated identity system (like EGI Check-in). In this case, HPC providers configure the synchronisation of the accounts with the federated LDAP registry that include among other user information, the public keys of the users. The SSH key access is currently used in the C-SCALE project for the integration of HPC providers[5], but it will not be tested for EGI-ACE.
- Access via middleware/portals/gateways. In this case, users do not get direct access via SSH, but a set of middleware components will handle the user access and the interaction with the HPC system. For EGI-ACE, we will consider the existing HTC middleware for delivering access to the providers. This is the simplest option to implement for the infrastructure as it would not require any additional change. However HTC middleware cannot be always deployed at the HPC systems due to policy restrictions. Access via web portals and gateways work in a similar way: a web accessible portal handles the submission of jobs to the HPC cluster and users do not have direct access to the underlying system. The ENES pilot will provide this kind of functionality for accessing HPC systems using a Jupyter notebooks interface.
- Delivery of virtual HPC infrastructures as a Service. In this case users can create their own custom HPC environment using virtual machines or bare-metal hosts that are spawned and configured to support the HPC applications. The use of virtualisation imposes certain overhead, but accelerator hardware (GPUs or low-latency network like Infiniband) can also be configured to minimise this overhead and provide an HPC-like performance on top of virtualised resources.

### 3.1.1 Provisioning and deprovisioning of accounts

Before a user can access a HPC installation, a user account needs to be available on the system. Accounts can be either created 'on-the-fly' as the user first accesses the system or be provisioned via some offline mechanism. The provisioning and deprovisioning of accounts normally follow strict policies and procedures that are very system-dependent, thus

[5]C-SCALE - D3.1 Initial Design of the Compute Federation https://zenodo.org/record/5084884

providing common solutions may impose undesirable requirements on the individual centres and clash with local policies and setups. Even without a common approach for the provisioning of accounts, it can be technically supported with LDAP registries like the one provided by Check-in[6] or using pool accounts as in most grid systems. Besides the provisioning of the accounts, federated users need to be mapped to these local accounts, which is determined by the configuration of the access mechanism to the system.

### 3.1.2 OIDC token based ssh access

OIDC (OpenID Connect) is an identity layer, built on top of OAuth protocol. It allows Clients to verify the identity of the End-User based on the authentication performed by an Authorization Server, as well as to obtain basic profile information about the End-User. This protocol reduces the friction experienced by the End-Users while allowing service providers to get the information they need relatively quickly and well-defined manner. EGI Check-in system is built on top of the same technology.

With the increasing requirement of accessing HPC systems from cloud environments and using these systems for fast and distributed processing, a need for using existing authentication systems for accessing HPC clusters became a necessity.

One of the ways for achieving this interconnection is utilization of a layer called SSH-OIDC[7]. SSH-OIDC adds capabilities for authenticating a user over SSH using its OIDC token obtained from any OpenID Connect supporting identity provider, including EGI Check-in.

SSH-OIDC is a relatively simple to install and administer system which contains two ends. Hence, it's installed both on client and server systems, and allows for creation of a side-channel to transfer OIDC related validation traffic and other side-tasks need to be handled during login. This side channel can be encrypted with TLS for enabling around-the-world, in-the-open secure deployment.

In general, SSH-OIDC has these outstanding features:

- Allows any user to login via any OpenID Connect provider.
    - Allows narrowing of acceptance via provider, Virtual Organization or both.
- Creates required users and groups automatically, assigns persistent usernames.
    - Groups users from the same roles under the same groups, allowing cooperation.
    - Allows username creation schemes to be configured.
- Installation of the components doesn't interfere with normal user authentication capabilities on either end.

The server side of SSH-OIDC contains two components. A service called motley-cue and an authentication plug-in called pam-ssh-oidc. Former module handles user mapping and related tasks, while the latter plugin allows tokens to be passed instead of passwords and be verified for allowing the user in.

---

[6] LDAP support in Check-in: https://docs.egi.eu/users/check-in/vos/#ldap

[7] https://github.com/EOSC-synergy/ssh-oidc

Similarly, the user side of the installation contains two components. A command line tool mccli for interfacing with motley-cue service on the server side and a oidc-agent tool for obtaining the tokens, and managing the accounts (hence tokens) on the client side, including encryption for increased security. After completing installation, a user might get its token from an OpenID Connect provider via the oidc-agent tool and connect to a supporting server via mccli tool.

Due to evolving nature of the software, an installation guide is not included in this handbook, however the following links will provide all the necessary information regarding to its installation and use:

- SSH-OIDC official document repository[8] (GitHub).
- Client quick installation guide[9] (kit.edu).

Since the software is distributed as RPM and DEB packages, installation and maintenance of the utilities via standard system management commands is possible, and straightforward.

### 3.1.3 Middleware access

One of the goals is to allow the exploitation of processing resources available in an existing HPC machine, e.g. by single-node (multicore) or single-core HTC jobs possibly by community specific workload management systems external to the HPC itself. This means that we expect that externally submitted jobs need to run on HPC provided resources. A typical workflow is the one where experiment specific pilot jobs reach the resource and call back the Experiments Workload Management Systems and receive payloads, which are executed inside a runtime environment.

In this scenario the HTCondorCE can represent a HPC edge node with access to the external IP ranges that can be carefully defined upfront. The edge node has the role then to submit to the internal batch system. The latter is a key aspect and here HTCondor represent a suitable solution because natively support the compatibility with server batch middleware among which SLURM which is a popular between HPC Centers.

Since one of the objectives of the work is to allow the exploitation of processing resources available in HPC clusters, e.g. by single-node (multicore) or single-core HTC jobs for data processing, it is natural to rely on the job router daemon capability natively built-in by HTCondor which is the key feature which allow to translate HTC submitted jobs into the internal batch. This provides the ability to transform vanilla jobs to the “slurm” batch type, thus allowing HTCondor to interface with Slurm and therefore supporting a mixing of HPC and HTC resources. Another key feature of the job routing daemon is to allow both automatic transformation as well as custom policies. Last but not least, this grants a high level of flexibility since, for example, one could use any python-based script for the implementation of a custom policy.

Another key aspect is that using such a HTCondor based approach opens the possibility even to the federation of distributed providers (either HPC or not) building a pool of heterogeneous resources.

---

[8] https://github.com/EOSC-synergy/ssh-oidc
[9] http://ssh-oidc-demo.data.kit.edu/

### 3.1.4 HPC as a Service

EGI operates a federated IaaS cloud that allows users to create virtual infrastructures where to run their workloads. Although these rely on hypervisors to create Virtual Machines, specialised hardware like GPU accelerators and low latency networks like Infiniband can be configured as PCI Passthrough devices without any significant performance overhead thus enabling the execution of certain HPC workloads. EC3 (Elastic Cloud Compute Cluster)[10] provides a tool to create elastic virtual clusters on Infrastructure as a Service (IaaS) providers, including support for SLURM and similar batch systems found in HPC systems. This approach allows for a very flexible and customizable environment for executing the user applications with complete control on the operating system and applications available and the associated hardware to each of the virtual nodes while keeping similar interfaces as those available in the HPC centres (SLURM). The ELI-NP pilot will explore this kind of solution and provide a detailed description in the next version of the document.

## 3.2 Operational integration

Having access using federated authentication and authorisation is the first step to integrate HPC centres in the EOSC Compute Platform. Operational integration enhances the federation from the operational perspective: they provide insights on the capacity consumption by users, simplify user interaction via a helpdesk, or provide service availability and reliability metrics.

### 3.2.1 Accounting

The Accounting service, provided by APEL[11], parses the logs from many computation platforms such as SLURM, OpenStack, OpenNebula and others, sends them to APEL, where they will be summarised, transformed and sent to the Accounting Portal for display. In the case of some large organisations, such as WLCG or OSG, those can do the summarization and send their records already processed.

AMS is used both for publication of non-summarized or summarised data, and for sending the final data to the Portal.

As part of the task, we evaluated the scenario of using the APEL software to publish an existing SLURM-based workload, in this case the parsing is done directly by the apel-parsers component instead of second-party parsers like cASO, which in many cases were developed to address the fragmentation existing in data representation between the existing Cloud provider management software.

We installed apel-parser successfully and configured it to point to our SLURM-powered computation node. The publication is done at fixed intervals , and in case of errors or gaps in the publication there are special commands that allow selective republication of the accounting data, although these must be used in tandem with APEL, so that old data is replaced seamlessly with the corrected/more complete data.

---

[10] https://docs.egi.eu/users/ec3/
[11] https://docs.egi.eu/internal/accounting/

In our case, the publication was successful, and in brief, the existing accounting solutions seem to work correctly, at least for our use case in a SLURM computing node.

### 3.2.2 Monitoring

Monitoring is key to gain insights into the status of an infrastructure. HPC providers in EGI-ACE already count with detailed monitoring to ensure the regular operation of the systems and detection of internal issues, thus this kind of monitoring will not be covered by the pilots. Instead, we will focus on ensuring the access interfaces of the EOSC Compute Platform are operational and available. ARGO[12] - the monitoring solution of EGI- can be extended with new probes and as part of the pilots we will implement those missing. For accessing the HPC providers using HTC middleware there are already an extensive set of probes available. In the case of ssh-oidc based access, the probes can leverage the existing monitoring credentials in ARGO to obtain Check-in tokens that can be used to access the system. A probe to test that the ssh interface is available (reachable) and that it can successfully be used for accessing the provider is currently under development and will be further detailed in the second version of the document.

### 3.2.3 Configuration Database

The EGI Configuration Database (GOCDB)[13] is a central registry that records topology information about all sites participating in the EGI infrastructure. As the HPC centres are already part of the infrastructure, there exists an entry in the GOCDB that lists all the relevant endpoints made available to users. New HPC providers entering the infrastructure need to follow EGI's PROC09[14] (Resource Centre Registration and Certification) that cover all the steps required for registering and certifying new Resource Centres (sites) in the EGI infrastructure. Certified Resource Centres make resources available to international user communities and guarantee a minimum quality of service of the resources (currently expressed in terms of monthly availability and reliability as obtained by the monitoring).

GOCDB also collects information about all the service endpoints available at each site. These endpoints are used by ARGO to automatically monitor and calculate the availability and reliability metrics depending on their service type (service types are pieces of software while service endpoints are a particular instance of that software running in a certain context).

For the HPC integration we plan to create new service types that cover the access mechanism tested in the pilots.

### 3.2.4 Helpdesk

The EGI Helpdesk[15] is the entry point and ticketing system/request tracker for issues concerning EGI services. New service providers can integrate into the Helpdesk by creating a dedicated support topic listed on the Helpdesk user interface (for users to ask questions

---

[12] https://docs.egi.eu/internal/monitoring/
[13] https://docs.egi.eu/internal/configuration-database/
[14] https://confluence.egi.eu/display/EGIPP/PROC09+Resource+Centre+Registration+and+Certification
https://confluence.egi.eu/display/EGIPP/PROC09+Resource+Centre+Registration+and+Certification
[15] https://docs.egi.eu/internal/helpdesk/

or raise issues directly to the provider). Resource centres registered in the Configuration Database will be automatically available in the Helpdesk for routing tickets as needed so no extra integration step is needed to be part of the Helpdesk.

## 3.3 Integration with the EOSC Marketplace

The integration with the EOSC Marketplace and handling the orders from providers is still under definition. There are two options ahead:

1. Individual HPC providers can be registered in the marketplace as shown in the screenshot below for CESGA:

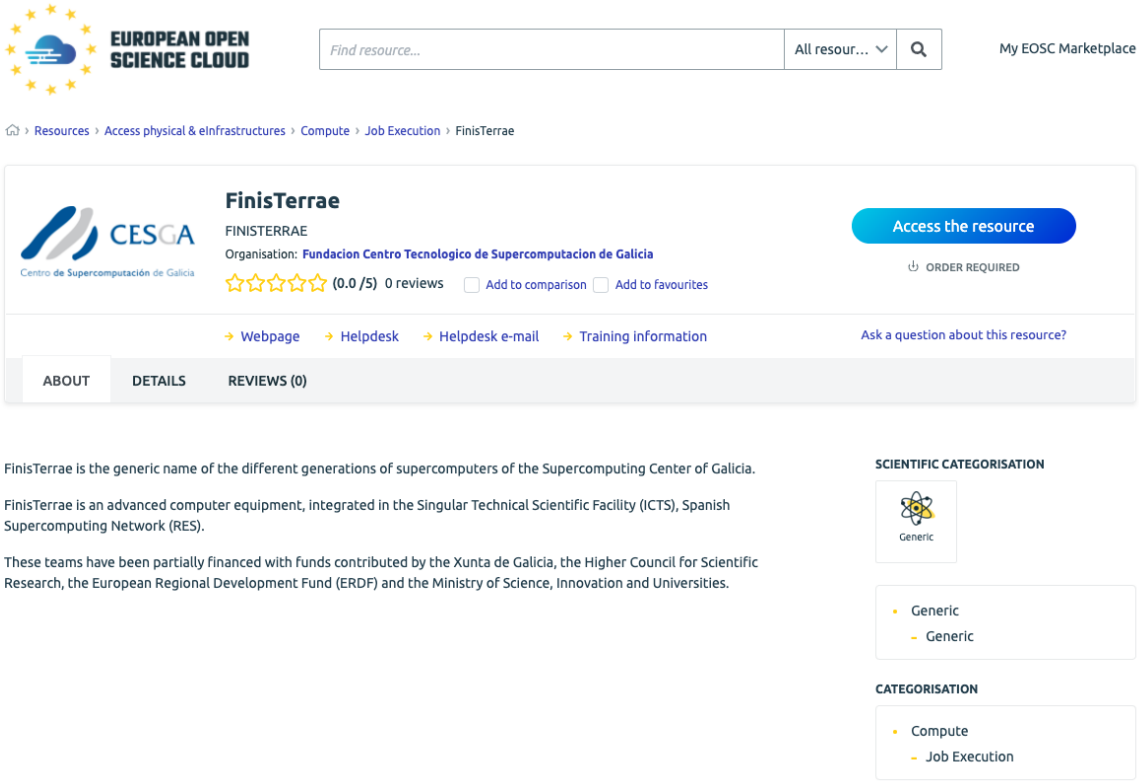


Fig 1: FinisTerrae - CESGA HPC - registered in EOSC Marketplace

2. A common entry for the EOSC Compute Platform that includes the HPC providers may be used in the future. (Similarly how EGI HTC, or EGI Cloud providers appear in the Marketplace.)

While the 1st option provides more flexibility and autonomy for providers, it would not offer the ‘one-stop-shop’ service for users, and these users have to still compare and select the most suitable HPC site themselves. In case of a single entry (2nd option) the resource selection and negotiation can be on the EOSC Compute Platform coordinators, moreover it can enable brokering of multiple types of resources from the Platform in a single transaction.

# 4 Application support

All pilots included in EGI-ACE plan to execute their workloads using containers. The preferred tool for supporting these containers is udocker. Data transfer pilots are currently ongoing and will be detailed in the upcoming version of the document.

## 4.1 udocker

udocker is a user-oriented tool to execute containers in user space without requiring root privileges. udocker enables basic download and execution of containers by non-privileged users in Linux systems. It can be used to access and execute docker containers in batch systems and interactive clusters that are managed by other entities such as grid infrastructures, HPC clusters or other externally managed batch or interactive systems. udocker is a wrapper around several tools and technologies to pull container images and execute them with minimal functionality. Since root privileges are not involved, most operations that require privileges will not work under udocker. This limitation does not affect user applications. udocker itself is written in Python and has a minimal set of dependencies so that it can be executed in a wide range of Linux systems.

Since udocker does not require any type of privileges nor the deployment of additional software by system administrators, it can be easily deployed by the end user in HPC clusters. Users can download udocker themselves and install the tool in their home directory.

Advantages of udocker:

- Can be deployed by the end-user
- Does not require privileges for installation
- Does not require privileges for execution
- Does not require compilation, just download udocker
- Encapsulates several tools and execution methods
- Includes the required tools already statically compiled to work across systems
- Provides a docker like command line interface
- Supports a subset of docker commands: search, pull, import, export, load, save, login, logout, create and run
- Allows loading of docker and OCI containers
- Supports NVIDIA GPGPU applications
- Can execute in systems and environments where Linux namespaces support is unavailable
- Runs both on new and older Linux distributions including CentOS 6, CentOS 7, CentOS 8, Ubuntu 14, Ubuntu 16, Ubuntu 18, Ubuntu 20, Ubuntu 21, Alpine, Fedora, etc

The basic flow for udocker usage is:

1. The user downloads udocker to its home directory and executes it
2. Upon the first execution udocker will download additional tools
3. Container images can be fetched from Docker Hub with udocker pull
4. Containers can be created from the images with udocker create
5. Containers can then be executed with udocker run

In addition and similarly to other container tools:

A. Containers can be loaded from file with udocker load -i
B. Tarballs can be imported with udocker import

Figure 2. provides an outline of using udocker to execute containers. Containers can be downloaded from a repository with pull, alternatively they can be loaded or imported from files. The container layers and metadata corresponding to the images are stored in the user home directory under $HOME/.udocker/layers. The content of these layers can then be extracted (flattening) to create a single file system tree for execution that is also placed under $HOME/.udocker/containers. Code within the extracted directory tree can then be executed using one of the supported execution modes.

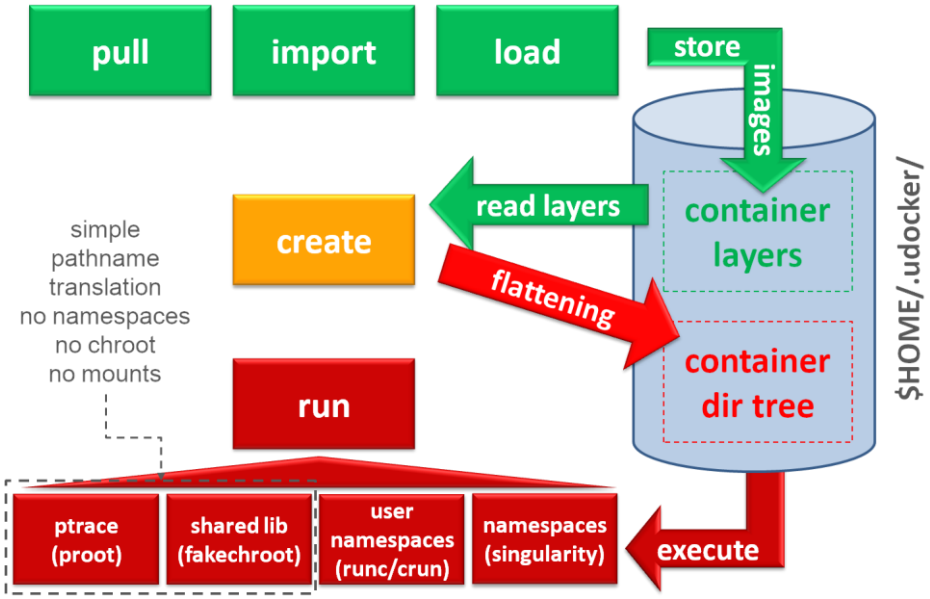


Fig 2: outline of containers execution with udocker

Installation can be performed using several methods. The installation from a release tarball is the recommended approach. Installation instructions for sites without outbound connectivity and additional information for shared installations are also available from the installation manual. Detailed installation instructions are available at: https://indigo-dc.github.io/udocker/installation_manual.html

The user manual including command reference and usage recommendations and considerations is available at: https://indigo-dc.github.io/udocker/user_manual.html. Information on the use of udocker with GPUs and MPI is also available in the user manual.

udocker “executes” the containers by simply providing a chroot like environment to the extracted container. udocker is meant to integrate several technologies and approaches hence providing an integrated environment that offers several execution options. For further information on selecting execution modes see the user manual section on udocker setup. The supported execution modes include:

| Execution mode | Tools | Description |
|---|---|---|
| P1 | PRoot | Accelerated mode using seccomp |
| P2 | PRoot | Same as P1 without seccomp accelerated mode |
| F1 | Fakechroot | Execution through loader invocation |
| F2 | Fakechroot | Same as F1 with modified loader to prevent loading from host |
| F3 | Fakechroot | Execution with fixed ELF headers for libraries and executables |
| F4 | Fakechroot | Same as F3 plus dynamical fixing of ELF headers |
| R1 | runc or crun | Rootless user mode namespaces |
| R2 | runc or crun | Same as R1 plus P1 for software installation |
| R3 | runc or crun | Same as R1 plus P2 for software installation |
| S1 | Singularity | Uses singularity if available in the host |

The Pn modes are the most generic and are used by default. The Fn modes are the fastest but require shared libraries matching the container libc, these libraries are provided with udocker for the most popular distributions. Both the Pn and Fn modes do not require the use of Linux namespaces. These modes intercept calls to the system and perform pathname translation to mimic a chroot environment.

Most of the required tools are provided with udocker already compiled and ready for use. However, the Sn mode requires the installation of the corresponding tool in the host system. The Rn modes require “user namespaces” support enabled in the Linux kernel.

Relevant links:

- Source code repository: https://github.com/indigo-dc/udocker
- Releases: https://github.com/indigo-dc/udocker/releases
- Complete documentation: https://indigo-dc.github.io/udocker

# 5 Conclusions

Thanks to the four EGI-ACE HPC pilots, the EOSC Compute Platform will be expanded to cover HPC providers to support computing workflows that make combined use of HTC, HPC and Cloud. In the first half of the piloting activity, the focus was given to provide access to the HPC systems using federated authentication and providing the use cases with the tools to run their containerised workload across different providers. In this document we report the various integration options and the main areas of work that will be covered by the pilots and completed by M18 of the project (June 2022). The handbook will be updated at the end of the piloting phase with a revised version that will expand the details on the available access mechanisms and those integration areas still under development (especially monitoring and presence in the EOSC marketplace). The data transfer aspects to enable cross-platform workflows will also be covered in the following version.