HPAC Platform including CEA resources (D7.5.1 - SGA2)

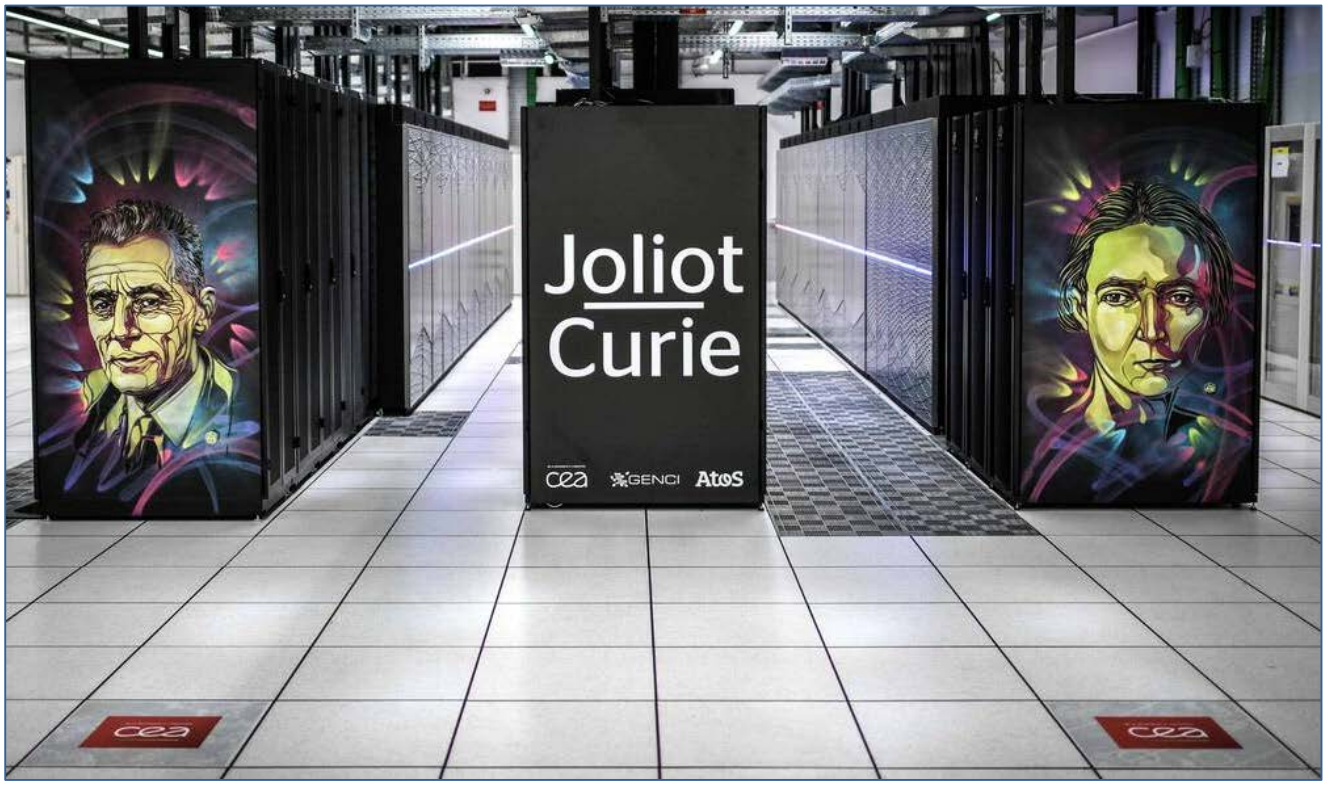

Figure 1: The Joliot Curie Tier-0 system in the computing centre TGCC at CEA.

The goal of this work is to integrate the compute and storage resources at CEA into the High-Performance Analytics and Computing Platform (HPAC) developed and operated by SP7.

Project Number:	785907	Project Title:	Human Brain Project SGA2

Document Title:	HPAC Platform including CEA resources
Document Filename:	D7.5.1 (D46.1 D85) SGA2 M8 ACCEPTED 200731.docx
Deliverable Number:	SGA2 D7.5.1 (D46.1, D85)
Deliverable Type:	Demonstrator
Work Packages:	WP7.5
Dissemination Level:	PU = Public
Planned Delivery Date:	SGA2 M8 / 30 Nov 2018
Actual Delivery Date:	SGA2 M23 / 24 Feb 2020; Accepted: 31 Jul 2020
Author(s):	Emmanuel PENOT, CEA (P11) Frederic COMBEAU, CEA (P11) Dorian KRAUSE, JUELICH (P20)
Compiled by:	Dorian KRAUSE, JUELICH (P20)
Contributor(s):	
SciTechCoord Review:	Marc MORGAN, EPFL (P1)
Editorial Review:	Annemieke MICHELS, EPFL (P1)
Description in GA:	The compute and storage resources of the new HPAC partner CEA are integrated into the HPAC Platform, including the corresponding documentation in the Guidebook. The implementation of a use case workflow is used to demonstrate the functionality and documented in a short report.
Abstract:	The Deliverable reviews the current status of the integration of the CEA resources into the HPAC Platform, details the identified challenges resulting in a delay of these activities and discusses the on-going activities and next steps.
Keywords:	HPAC, High Performance Analytics and Computing, HPC
Target Users/Readers:	Computational neuroscience community, computer scientists, consortium members, HPC community, neuroscientific community

Table of Contents

Table of Figures

1. Introduction

The High Performance Analytics and Computing (HPAC) Platform combines compute and storage resources from major European HPC sites: BSC in Spain, Aachen in Germany, JUELICH in Germany, CSCS in Switzerland and CINECA in Italy. In SGA2, CEA in France joined as partner and agreed to integrate its state-of-the art compute infrastructure at the Très Grand Centre de Calcul (TGCC) data centre in Bruyères-Le-Châtel into the HPAC.

The work on the integration of CEA into the HPAC has been performed in the context of Task T7.5.1 since the beginning of SGA2. The initial analysis of the infrastructure and security aspects of the integration of CEA into HPAC has revealed that the CEA integration into the Platform is significantly more complicated than anticipated in the SGA2 preparation. For this reason, the initial Deliverable D7.5.1 for the integration of CEA into the HPAC Platform could not be met. This is primarily due to the strong legal security obligations under which CEA operates which surpass the requirements and external constraints of all other HPAC sites.

The security infrastructure at CEA is centred on the Kerberos network authentication protocol. Kerberos enables secure client/server authentication over unprotected networks within one security realm. At the core of the Kerberos authentication infrastructure is the management of tokens that are secured by a user secret, usually the user password. Such tokens, specifically the so-called tgt ("token granting tokens", often also referred to as "ticket granting tickets"), can be used to request – without further password-based authentication in a "single sign on" fashion – access to specific services (e.g. LDAP or HTTP) and resources (e.g. NFS file systems). The services and resources themselves need to support Kerberos ("kerberized" services).

This high-level Kerberos introduction already highlights the central difficulty with the integration of a Kerberos infrastructure with a federated service, such as the UNICORE service underlying the HPAC platform. Kerberos requires a locally provided user secret (password) for the encryption. Federated services on the other hand rely on remote authentication services and do not foresee the availability of user secrets for access to remote sites.

Two HPAC-relevant sites employ Kerberos: CSCS and CEA. However, CSCS does not deploy "kerberized" services but relies solely on Kerberos for the login procedure. Therefore, the support of UNICORE at CSCS has not been an obstacle in the setup of the HPAC Platform. CEA on the other hand employs a large number of "kerberized" services and strives for full Kerberos-enablement of the infrastructure. As mentioned before, this is not solely a technical or architectural choice but strongly driven by CEA's strong legal security obligations. None of the other HPAC sites is subject to similar external constraints.

In addition to the requirement to fully support the Kerberos-centred security infrastructure at CEA, legal constraints require a local authentication step at CEA (i.e. against a service hosted on a server in the TGCC premise) for all users before access to the resources can be granted.

In Task T7.5.1, CEA has developed technical solutions addressing these two challenges that enable a functional and compliant integration into the Platform in such a way that the overhead for users is kept to a minimum. The technical solution has taken into account future requirements, in particular the migration of the HPAC platform into the Fenix infrastructure currently built up in the context of the ICEI project[1].

Deliverable D7.5.1 describes the technical solution implemented for the integration of CEA into HPAC and demonstrates the functionality by means of a sample use case workflow. In addition, the documentation in the HBP High-Performance Analytics & Computing Platform Guidebook[2] has been updated to reflect the new infrastructure components.

[1] Interactive Computing e-Infrastructure (ICEI), another SGA under the HBP FPA building the Fenix infrastructure

[2] https://hbp-hpc-platform.fz-juelich.de/?page_id=2212

2. Details about the Technical Implementation

CEA has been using Kerberos to secure service access using the "single sign-on" paradigm in its supercomputing centres since more than 10 years. To cope with some HPAC specificities, and particularly batch executions, CEA had to introduce a component dedicated to Kerberos tokens caching, renewal and ACL-based provisioning. This open-source component, called "auks", is publicly available on GitHub since 2009.

The usual way users are accessing supercomputing at CEA is through SSH connections to supercomputer login nodes. SSH connections, whether based on x509 certificates or traditional login/password challenges, result in users having a valid Kerberos credential in their login node session. This allows users to transparently access kerberized services like LDAP, NFSv4 file systems, other nodes using SSH, ...

Compute nodes inherit the same security restrictions as login nodes and users need to have valid Kerberos credentials to access kerberized services from them. Without a valid Kerberos token, compute nodes cannot be used by users.

Auks helps batch systems to securely transfer Kerberos credentials from login to compute nodes on behalf of the users. It enables users to push their Kerberos credential to a central secured place, called auks repository, right before submitting their jobs. Using Slurm, as is the case at CEA, this process can be fully automated using a dedicated plugin. Once jobs are executed, the privileged batch system daemons running on the compute nodes can retrieve users' Kerberos credentials and set them up in the job environment and let users access mandatory services for their computation.

Having a valid Kerberos token is the major technical constraint for being able to use CEA compute and storage resources.

An additional constraint is due to the current CEA security policy. To be able to access the TGCC infrastructure, a user must be authenticated using a service hosted at TGCC and administered by its IT team. This means that independently of whether an authentication has already been performed externally (e.g. against the HBP OIDC server), users need to be authenticated at the CEA entrance.

These two constraints must be met in order to provide access into the CEA Joliot-Curie machine from the HPAC platform.

After having successfully deployed and configured UNICORE for its latest supercomputer access, CEA had to find a way to respect these two constraints:

- ensure that users are authenticated against the TGCC Identity Provider (IdP) while using UNICORE,
- ensure that users have a valid Kerberos credential while submitting their jobs.

The logic used to respect the constraints was to create and add a web authentication gateway, that we called here a "Kerberos web server", associated to CEA's internal Kerberos key distribution centre (KDC). This Kerberos web server aims at authenticating users wishing to access UNICORE. The solution is based on KFS, developed at the CEA and available on Github (https://github.com/cea-hpc/kfs). During the authentication stage of a user, a valid Kerberos token is acquired if the authentication is successful. This token is pushed to a HPAC-dedicated auks repository. It is then retrieved by UNICORE (more technically by the UNICORE daemon running on login node) before accessing the resources on the targeted supercomputer.

Users trying to access CEA resources using UNICORE without having a valid connection to the web authentication gateway will fail to provide a valid Kerberos token and will therefore be denied access to the resources and to the launch of jobs.

The following sequence diagram (Figure 2) illustrates the interaction between a user and the various components while launching a job on a CEA supercomputer with this approach.

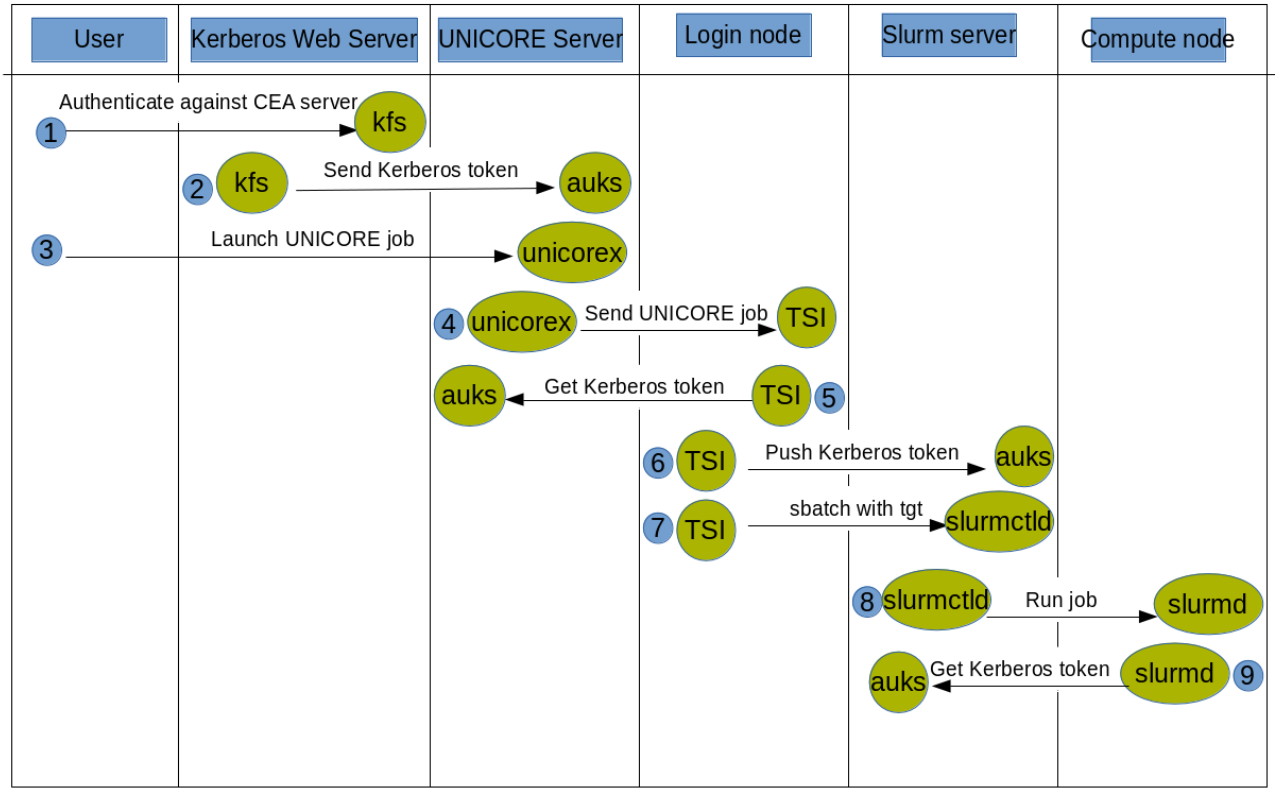

Figure 2: Schematics of the workflow for accessing the HPAC CEA resources with UNICORE.

Note that two auks repositories are used in the approach. The legacy one, located on the Slurm server, enables users connected on login nodes to use the compute nodes using Slurm. The newly introduced one, on the UNICOREX server, enables users to access kerberized services on the TSI node and submit jobs when a valid token has been previously pushed into it using the Kerberos web server. This separation ensures that users connected to login nodes and accessing compute nodes do not automatically provide all the requested material for a proper access to the resources using UNICORE: an additional authentication is mandatory for that, in line with the legal obligations of CEA.

3. Description of a Use Case Workflow

In the following, a use case workflow is described in detail to illustrate the steps required to use the CEA HPAC resources from the HBP Collaboratory. The demonstrator notebook[3] is also available in the Collaboratory for review and test execution at the URL

https://collab.humanbrainproject.eu/#/collab/264/nav/1975?state=uuid%3D0503eae6-6edd-4863-8646-390b6ce22ad4 .

To highlight the seamless integration with other HPAC resources, the workflow encompasses resources from CEA as well as another HPAC site. For the latter, JUELICH was chosen, but any other site would be equally well suited.

1.1 Use Case Workflow

The workflow consists of the following steps:

1) Authenticate against the Kerberos web server;

[3] The notebook and the workflow described may differ in minor implementation details.

2) Launch jobs on the compute resources to interact with the storage resources.
   a) Calculate the MD5 hash value of a file on a Joliot-Curie login node;
   b) Transfer the file to the storage resources in JUELICH via the JUWELS supercomputer;
   c) Verify the successful transfer by calculating the MD5 hash value at the destination and comparing it with the initial hash sum.

1.2 Authentication against the Kerberos web servers

As a first step, the user must connect to the Kerberos web server for local CEA authentication (available at https://mistral.ccc.cea.fr) and enter their TGCC login and password.

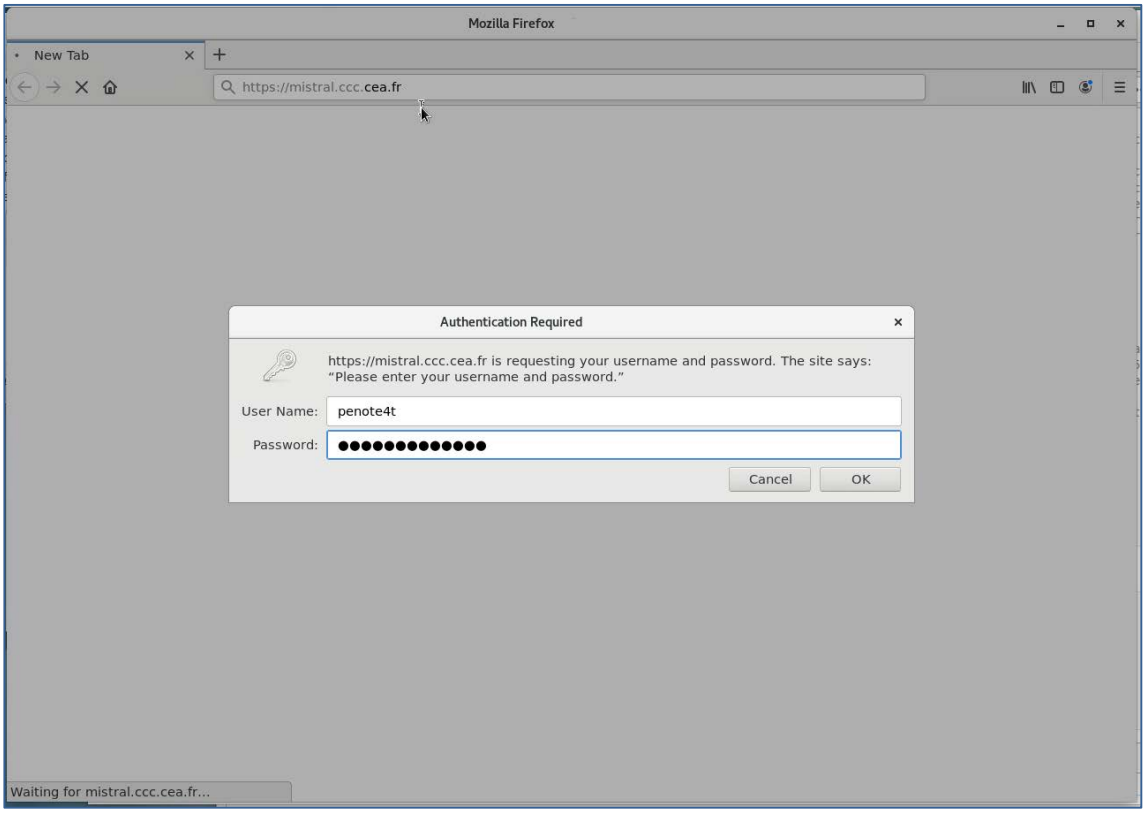

Figure 3: Kerberos web server interface for the local authentication.

If authentication is successful, the web page shows how long the user token is valid.

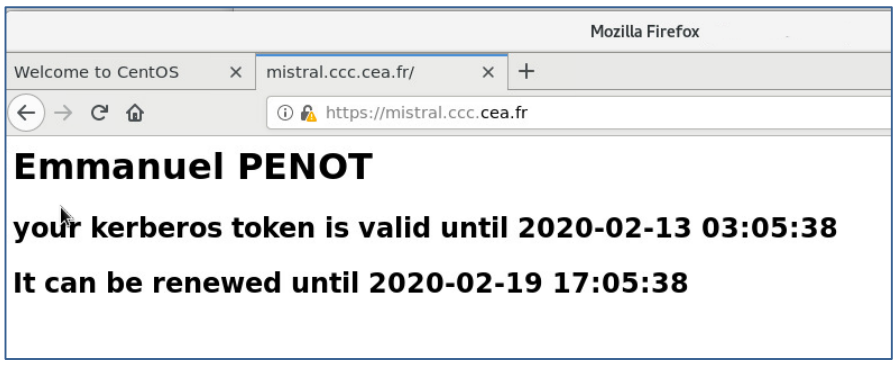

Figure 4: Kerberos web server page after successful authentication.

The validity period of the token is 10 hours. The auks server renews the token automatically during a 7-day period. Users can renew the token on the web server to extend the validity with the same process.

1.3 Job launch

Once authenticated, the user can make submissions to:

https://hpb-unicore.ccc.cea.fr/irene/rest/core

1.3.1 Calculate the MD5 hash value on Joliot-Curie

As the first step in this sample workflow, we check that we are successfully able to access the UNICORE server at CEA from the HBP Collaboratory:

```
In [54]: # replace the URL by the correct one...
         url = "https://hbp-unicore.ccc.cea.fr/irene/rest/core"

         auth = oauth.get_token()
         tr = unicore_client.Transport(auth)
         client = unicore_client.Client(tr, url)
         print(json.dumps(client.access_info(), indent=2))

         {
           "dn": "CN=Emmanuel Penot 308102,O=HBP",
           "xlogin": {
             "availableUIDs": [
               "penote4t"
             ],
             "UID": "penote4t",
             "availableGroups": []
           },
           "role": {
             "availableRoles": [
               "user"
             ],
             "selected": "user"
           }
         }
```

Figure 5: Verification of access in the HBP Collaboratory.

Afterwards, we calculate the MD5 hash value of the test file by executing the standard Linux utility md5sum on the CEA infrastructure:

```
In [ ]: jobs = {'Executable': "/usr/bin/md5sum", 'Arguments' :["$HOME/data_1.txt"], }
        url = "https://hbp-unicore.ccc.cea.fr/irene/rest/core/"
        auth = oauth.get_token()
        client = unicore_client.Client(tr, url)
        Myjob = client.new_job(jobs)
```

Figure 6: Calculation of the MD5 hash value on Joliot-Curie.

```
In [34]: wdsc = Myjob.working_dir
         wdsc.properties
         wdsc.stat("/stdout").raw().readlines()

Out[34]: ['4dbb1c99a07e922b3bccebe33bb460cd  data_1.txt\n']
```

Figure 7: Result of the MD5 hash value on Joliot-Curie.

1.3.2 File transfer

Next, we perform a copy from CEA to JUELICH via the HBP Collaboratory:

```
In [ ]: jobs = {'Executable': "/usr/bin/scp", 'Arguments' :["$HOME/data_1.txt", "juwels:"], }
        url = "https://hbp-unicore.ccc.cea.fr/irene/rest/core/"
        auth = oauth.get_token()
        client = unicore_client.Client(tr, url)
        Myjob = client.new_job(jobs)
```

Figure 8: Data transfer from CEA to JUELICH via the HBP Collaboratory.

For the sake of completeness, we verify the MD5 hash value manually on the target system.

```
[penot1@judac03 ~]$ ls -l
total 2
lrwxrwxrwx 1 penot1 jusers   9 Nov 20 10:27 shared -> ../shared
-rw-r----- 1 penot1 jusers 659 Jan 17 17:01 data_1.txt
```

1.3.3 Verification of the MD5 hash value on the destination system

Finally, we verify that the data transfer was successful by computing the MD5 hash sum on JUWELS.

```
In [21]: jobs = {'Executable': "/usr/bin/md5sum", 'Arguments' :["$HOME/data_1.txt"], }
         url = "https://zam2125.zam.kfa-juelich.de:9112/JUWELS/rest/core"
         auth = oauth.get_token()
         client = unicore_client.Client(tr, url)
         Myjob = client.new_job(jobs)
```

Figure 9: Calculation of the MD5 hash value on JUWELS.

```
In [13]: client = unicore_client.Client(tr, url)
         Myjob = client.new_job(my_job, inputs=local_inputs)

In [14]: wd = Myjob.working_dir
         wd.properties
         wd.stat("/stdout").raw().readlines()

Out[14]: ['4dbb1c99a07e922b3bccebe33bb460cd  /p/home/jusers/penot1/juwels/data_1.txt\n']
```

Figure 10: Result of the MD5 hash value on JUWELS.

4. Conclusion

CEA has developed and implemented a technical solution to make the CEA infrastructure accessible within the HPAC infrastructure, in particular to be able to launch UNICORE jobs, in such a way that the local authentication requirement is met and a valid Kerberos token is available for the UNICORE job to utilise the CEA file system and batch system.

This solution enables HPAC users to access CEA resources in a similar fashion as other HPAC sites, albeit with an additional authentication step as shown in Section 3.

However, CEA is planning enhancements to the setup. In particular, the next project step is to select and validate an identity provider (IdP) for the authentication step together with the security officer for authentication and Kerberos token management. CEA has started evaluating Keycloak for this goal. The solution must have the ability to handle Kerberos tokens, and in the future, be able to integrate into the ICEI AAI environment.

The description of the HPAC CEA storage and compute resources is available in the HBP High-Performance Analytics & Computing Platform Guidebook[4] and will be maintained and regularly updated there. At this point, only active data repositories[5] are provided as storage resources. CEA has recently finalised the procurement of infrastructure components for the Fenix e-infrastructure. One of the components is the SWIFT object storage. The deployment is executed in the context of the ICEI project and the resources will be available for the EBRAINS platform in SGA3.

[4] https://hbp-hpc-platform.fz-juelich.de/?page_id=2212

[5] Using the nomenclature of the Fenix infrastructure and the ICEI project, an active data repository is a site-local, typically performance-optimised, storage layer with a (near-)POSIX compliant interface. It differs from archival data repositories, which are accessible via the SWIFT object storage API.