EPJ Web of Conferences **245**, 07023 (2020) https://doi.org/10.1051/epjconf/202024507023
CHEP 2019

Abstracting container technologies and transfer mechanisms in the Scalable CyberInfrastructure for Artificial Intelligence and Likelihood Free Inference (SCAILFIN) project

Kenyi Hurtado Anampa[1,*], *Cody* Kankel[1,**], *Mike* Hildreth[1,***], *Paul* Brenner[1,****], *Irena* Johnson[1,†], *Scott* Hampton[1,‡], and *Tibor* Simko[2,§]

[1]University of Notre Dame, Notre Dame, IN, USA

[2]CERN, Geneva, Switzerland

Abstract. High Performance Computing (HPC) facilities provide vast computational power and storage, but generally work on fixed environments designed to address the most common software needs locally, making it challenging for users to bring their own software. To overcome this issue, most HPC facilities have added support for HPC friendly container technologies such as Shifter, Singularity, or Charliecloud. These different container technologies are all compatible with the more popular Docker containers, however the implementation and use of said containers is different for each HPC friendly container technology. These usage differences can make it difficult for an end user to easily submit and utilize different HPC sites without making adjustments to their workflows and software. This issue is exacerbated when attempting to utilize workflow management software between different sites with differing container technologies.

The SCAILFIN project aims to develop and deploy artificial intelligence (AI) and likelihood-free inference (LFI) techniques and software using scalable cyberinfrastructure (CI) that span multiple sites. The project has extended the CERN-based REANA framework, a platform designed to enable analysis reusability, and reproducibility while supporting different workflow engine languages, in order to support submission to different HPC facilities. The work presented here focuses on the development of an abstraction layer that allows the support of different container technologies and different transfer protocols for files and directories between the HPC facility and the REANA cluster edge service from the user's workflow application.

*e-mail: khurtado@nd.edu
**e-mail: ckankel@nd.edu
***e-mail: mhildret@nd.edu
****e-mail: pbrenne1@nd.edu
†e-mail: ijohnso1@nd.edu
‡e-mail: shampton@nd.edu
§e-mail: tibor.simko@cern.ch

1 Introduction

The REANA framework [1], used by the SCAILFIN project [2] to create analysis workflows that are easily reproducible, works on top of Kubernetes in order to orchestrate all service components and the scheduling of workers to run the payloads. This assumes support of docker and a shared file system between services and workers (see Figure 1). While the REANA team is working on supporting different submission backends in the framework (HTCondor [3] and SLURM [4] at present), these are currently focused on working with CERN resources [5]. The SCAILFIN project on the other hand, focuses on integrating REANA with different HPC environments.

In order to do this in SCAILFIN, we have created a submission backend integrated with VC3 [6], leveraging the authentication to the HPC submit nodes and proper translation of jobs from HTCondor to the different HPC supported batch systems, as seen in Figure 2.

A description of the mechanisms used to support different HPC container technologies and handle the transfer of files between the workers and the REANA cluster edge service by creating a user-level distributed file system will be presented here.

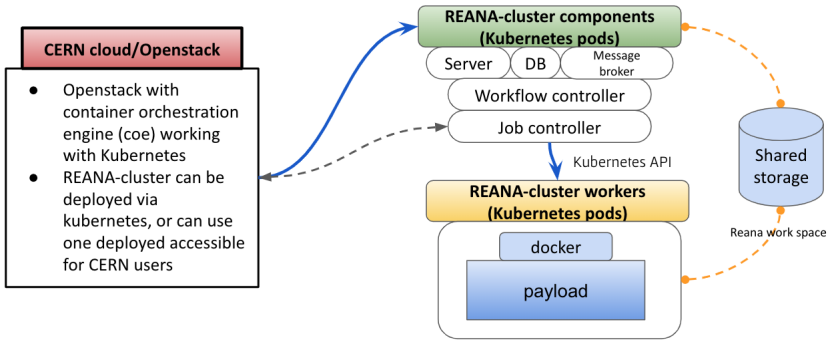

Figure 1. REANA cluster deployment via Kubernetes

2 Detecting and supporting different container technologies

Users can interact with the REANA cluster through a python-based client package. This client allows the user to create workflows, upload files to the workflow area and define the docker container images needed to run each step in the workflow (see Figure 3).

Docker support is normally not provided by HPC centers, mostly due to the root daemon privilege mode in docker, which imposes a security risk on multi-user facilities such as these. Instead, HPC friendly container technology solutions are frequently provided, such as Singularity [7], Shifter [8] or Charlie Cloud [9], which are compatible with docker images, but run the containers in unprivileged mode.

In order to support different container technologies, the job controller submission backend constructs the job submission object and passes the docker image as a job argument(see Figure 4). Subsequently, a job wrapper running in the worker node detects the container technology available (Singularity and Shifter are supported at present) and launches the container image with the proper parameters, as shown in Figure 5.

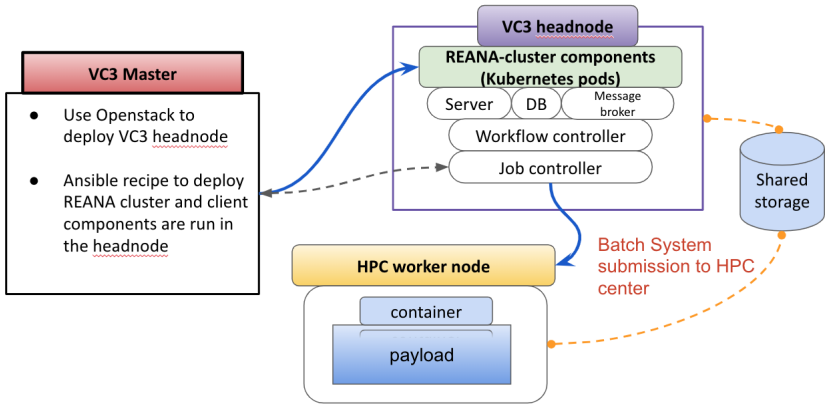

Figure 2. REANA cluster deployment via VC3

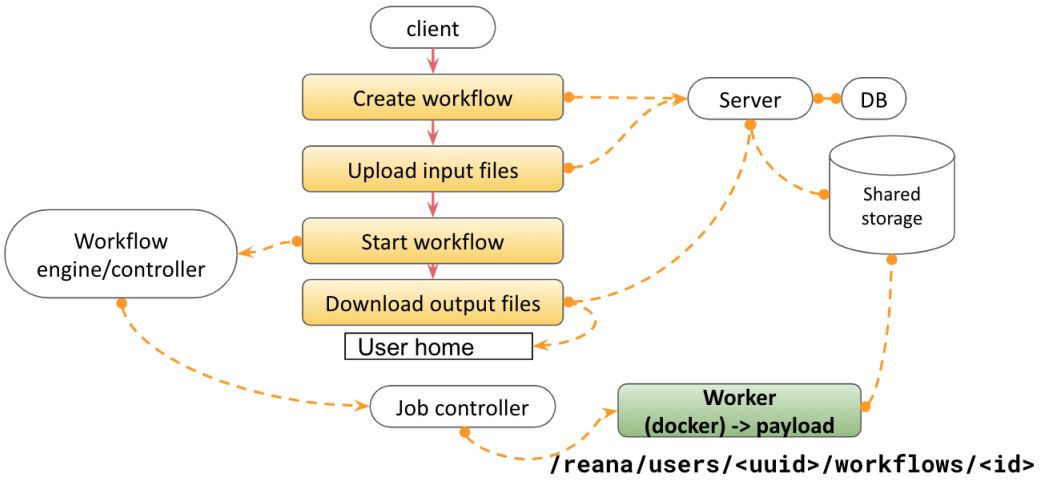

Figure 3. Schematic showing the REANA client creating a new workflow, uploading the necessary input files to a shared file system and running the payload inside docker containers.

This allows workflows to run on multiple HPC facilities supporting container technologies, without involving the user in providing the specific parameters per container technology for launching the container images.

3 Data transfer management

Input files uploaded by the users through the client are stored in a file system at the edge service by the server. This file system is shared between the REANA service components and the workers when Kubernetes is also used as the scheduler for jobs. With HPC facilities, jobs are submitted using their batch systems instead. Even though the REANA server can upload files to this file system area, HPC workers do not have access to them in this case.

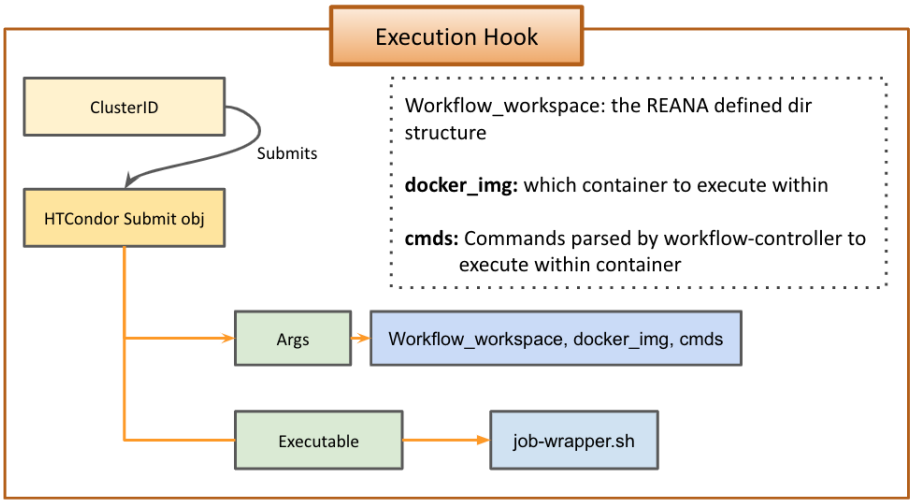

Figure 4. Job submit object creation and definition of arguments: workflow space absolute path, docker images and the commands to be executed

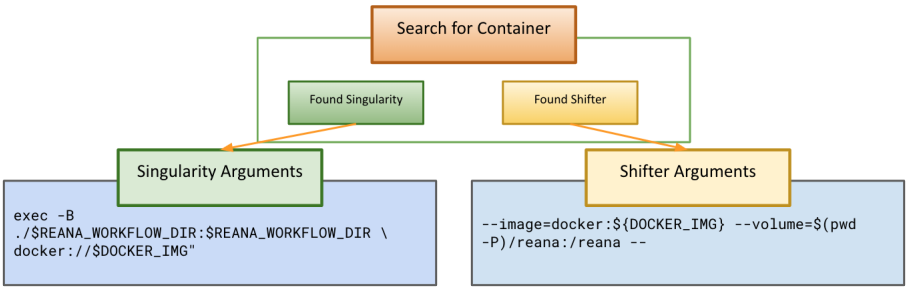

Figure 5. Detection of container technology available in the HPC worker and definition of parameters for it

To address this issue, SCAILFIN uses chirp [10], a user-level file system for collaboration across distributed systems such as clusters, clouds and grids without any special privileges. Chirp is integrated with HTCondor; a chirp server is automatically launched when the classad `+WantIOProxy=True` is added to job submit object. This server can then be used to share files between the VC3-headnode and the workers. This mechanism also takes care of the authentication using cookies created by HTCondor itself when this option is enabled.

In order to access this file system from the worker nodes, parrot [11] is installed using the vc3-builder [12] on the machine. Parrot is used to interact with the chirp server created by HTCondor; while HTCondor has its own client, parrot allows recursive access of directories on the fly, as opposed to the HTCondor chirp client.

As shown in Figure 6, parrot transfers all input files and directories for the job inside the job scratch area. The directory is bound inside the container, using the same absolute path

kubernetes workers would normally see these files at in the shared file system, in order to guarantee compatibility between both submission backends.

Finally, Figure 7 summarizes the main differences compared to the original infrastructure in Figure 3, using the chirp server for the workers to connect to the file system the REANA server writes to, as well as detecting and using Singularity or Shifter accordingly, to provide the proper environment for the job.

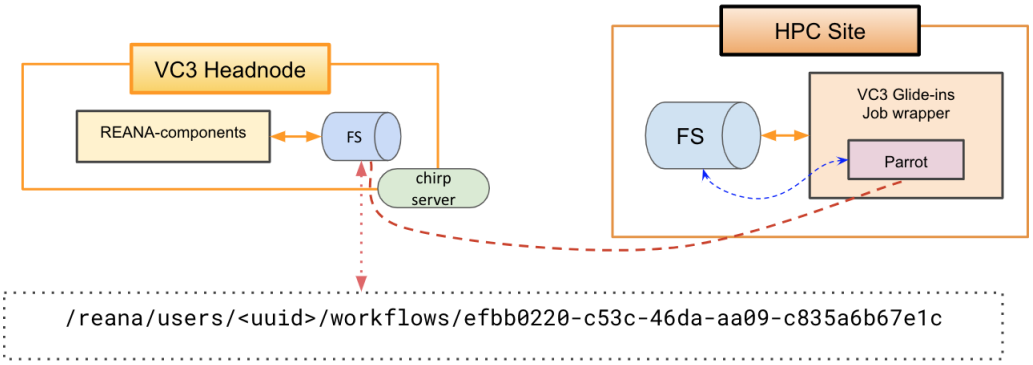

Figure 6. File management via Chirp and Parrot

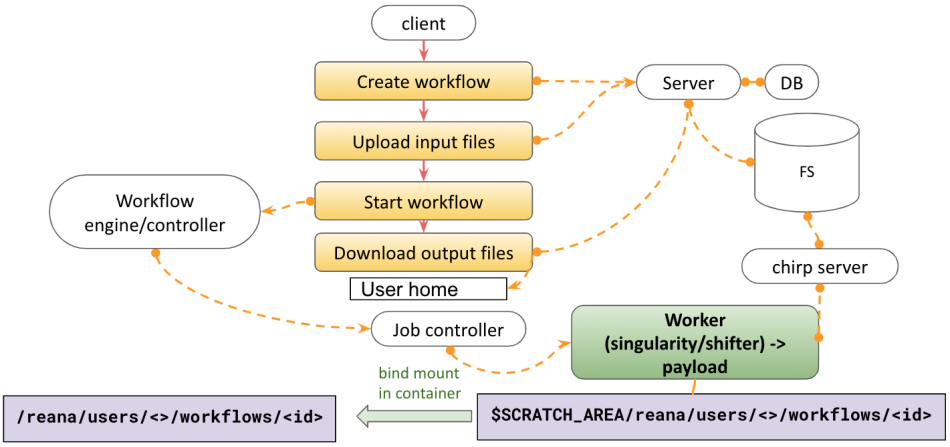

Figure 7. Schematic showing the REANA client creating and running new workflow, after integrating the detection and usage of HPC friendly container technologies and connecting the edge service file system used by REANA server via chirp and parrot

4 Conclusions

The SCAILFIN project has implemented mechanisms to automatically detect container technologies supported by HPC centers on the fly and use them in such a way that no modifications in the workflow definitions are necessary. Files are also shared between the HPC

facility workers and the REANA cluster edge service in a seamless way by using a user-level file system integrated with HTCondor, which can be used independently of the actual HPC facility supported batch scheduler.

Acknowledgement

The SCAILFIN project is funded by grants from the National Science Foundation (OAC-1841456, OAC-1841471, OAC-1841448). We are grateful to the VC3 project and the University of Chicago for the infrastructure provided for this work.

References

[1] T Simko et al., Search for Computational Workflow Synergies in Reproducible Research Data Analyses in Particle Physics and Life Sciences, IEEE 14th International Conference on e-Science (e-Science), Amsterdam, pp. 403-404 (2018)

[2] M Hildreth, K Hurtado Anampa, C Kankel, S Hampton, P Brenner, I Johnson, T Simko, Large-scale HPC deployment of Scalable CyberInfrastructure for Artificial Intelligence and Likelihood Free Inference (SCAILFIN), these proceedings (2019)

[3] D Thain, T Tannenbaum and M Livny, Condor and the Grid, *Grid Computing: Making The Global Infrastructure a Reality*, ISBN: 0-470-85319-0 (John Wiley & Sons, 2003)

[4] M Jette, A Yoo, M Grondona , SLURM: Simple linux utility for resource management, Lecture Notes in Computer Science 10.1007/10968987_3 (2003)

[5] M Rokas, P Brenner, S Hampton, M Hildreth, K Hurtado Anampa, I Johnson, C Kankel, J Okraska, D Rodriguez Rodriguez, T Simko, Support for HTCondor high-throughput computing workflows in the REANA reusable analysis platform, 15th eScience IEEE International Conference, San Diego, United States, 24 - 27 Sep 2019, pp.eScience (2019)

[6] L Bryant, J Van, B Riedel, R Gardner, J Caballero Bejar, J Hover, B Tovar, K Hurtado and D Thain, VC3: A Virtual Cluster Service for Community Computation, PEARC, **30**, 1-8 (2018)

[7] GM Kurtzer, V Sochat, MW Bauer, Singularity: Scientific containers for mobility of compute, PLoS ONE 12(5): e0177459. (2017)

[8] L Gerhardt, et. al., Shifter: Containers for HPC, Journal of Physics: Conference Series. 898. 082021. 10.1088/1742-6596/898/8/082021 (2017).

[9] T Priedhorsky, T Randles, Charliecloud: unprivileged containers for user-defined software stacks in HPC, Conference: the International Conference for High Performance Computing, Networking, Storage and Analysis 1-10. 10.1145/3126908.3126925 (2017)

[10] T Kosar, D Thain, M Albrecht, H Bui, Hoang, P Bui, R Carmichael, S Emrich, P Flynn, Data Intensive Computing with Clustered Chirp Servers, Data Intensive Distributed Computing, pp.140-154 DOI:10.4018/978-1-61520-971-2.ch006. (2012)

[11] D Thain and M Livny, Parrot: An application environment for dataintensive computing, Scalable Computing: Practice and Experience, 6(3):9–18, (2005)

[12] B Tovar, N Hazekamp, N Kremer-Herman, D Thain, Automatic Dependency Management for Scientific Applications on Clusters, IEEE International Conference on Cloud Engineering (IC2E) (2018)