Federation of compute resources available to the German CMS community

Federation of compute resources available to the German CMS community

R F von Cube, M Giffels, C Heidecker, G Quast, M B Sauter and M J Schnepf

KIT – Karlsruhe Institute of Technology, Germany

E-mail: ralf.florian.von.cube@cern.ch

Abstract. The German CMS community (DCMS) as a whole can benefit from the various compute resources, available to its different institutes. While Grid-enabled and National Analysis Facility resources are usually shared within the community, local and recently enabled opportunistic resources like HPC centers and cloud resources are not. Furthermore, there is no shared submission infrastructure available.

Via HTCondor's [1] mechanisms to connect resource pools, several remote pools can be connected transparently to the users and therefore used more efficiently by a multitude of user groups. In addition to the statically provisioned resources, also dynamically allocated resources from external cloud providers as well as HPC centers can be integrated. However, the usage of such dynamically allocated resources gives rise to additional complexity. Constraints on access policies of the resources, as well as workflow necessities have to be taken care of. To maintain a well-defined and reliable runtime environment on each resource, virtualization and containerization technologies such as virtual machines, Docker, and Singularity, are used.

1. Introduction

Recent surveys show that a significant amount of computing resources is locally available to institutes of the German high energy physics (HEP) community, however not shared among the national collaborators. Commissioning and federating such resources will enable us to mitigate the resource shortage in future LHC runs, improve the overall runtime of workflows and increase the usage efficiency of the resources.

Those resources, combined with the official WLCG [2] resources constitute a rather heterogeneous, hence challenging environment for administrators, as well as for users: For each resource there might be different access and usage policies, authorization and thereby authentication has to be handled, and the users expect a certain runtime environment in order to execute HEP workflows.

2. HTCondor

Computing payloads are organized as jobs, which consist of an executable and possible input and output files, and these jobs are scheduled using batch systems. Due to its reliability and scalability, HTCondor [1] is a very popular choice in HEP. A major part of the German WLCG resources and many other institutions are already using HTCondor as their batch system. Users submit jobs to their local HTCondor instance, which then checks within the local pool for resources which match the requirements. The requirements are defined by the user in the jobs

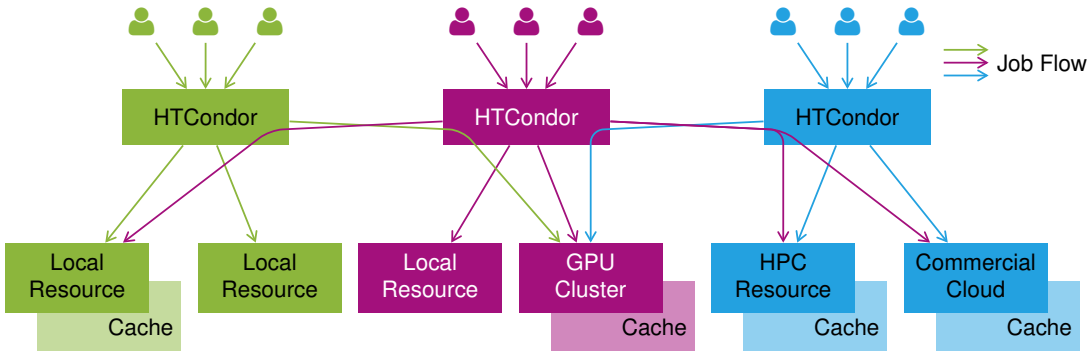

Figure 1. Federating resources from different providers with HTCondor's flocking mechanism allows users from different groups to exploit unused resources, and profit from dedicated hardware, as e.g. high throughput nodes with caching infrastructure, e.g. as described in [5], or GPU clusters.

submission file. If HTCondor finds a match, it takes care of transferring the executable and the corresponding input files to the respective resource. Then the job gets executed on this resource and after finishing, the relevant output files get transferred back.

HTCondor enables automatic execution of jobs in Docker [3] or Singularity [4] containers. This facilitates deploying the correct runtime environment for each job, no matter on which resource it is run, as it permits choosing the operating system and supplied software. The containerization of the jobs is done completely by HTCondor and thus transparent to the user. This allows to define default container images on each resource, while the user can still opt for a different image.

HTCondor also offers a so-called flocking mechanism in order to send jobs to remote resource pools. If configured, jobs for which no match within the local resource pool can be found become eligible for flocking. Those jobs then get sent to a configured remote resource pool and run through its match making process. The administrator of the remote resource pool can configure, how the jobs flocking into the pool are handled. HTCondor takes care of the authentication and, if a match is found within this pool and the jobs starts running, the file transfers to the matched resource. With this mechanism, multiple pools can be connected and jobs can move relatively freely to remote resources in order to find a suitable match. Figure 1 depicts an example.

3. Federating resources

Many groups have access not only to local resources, like desktops or local computing clusters within their institutes, but also to HPC clusters, cloud computing infrastructure, or more general other remote computing clusters, which we call opportunistic resources. Using the flocking mechanism, jobs can be sent to remote resource pools transparently to the user. By integrating opportunistic resources available to a group into the local resource pool with the pilot concept, as described in [6] and connecting the pools to one national pool, the overall utilization of the resources can be optimized, as described in section 3.1. The German HEP infrastructure is an ideal candidate for applying such concepts, as HTCondor is already widely used, the funding structure is similar, and collaborations within the German HEP community are already established. However, this approach can be easily extended to further communities as well. Flocking can take place on different-level computing resources as described in the following.

3.1. Federating resources for end-users

The demand for resources within an institute varies heavily over time. By flocking several institute's HTCondor pools this high variation can be mitigated, resulting in 1. a more efficient usage of the available resources and 2. an overall shorter processing time. Another advantage is to be able to use dedicated hardware of other groups: For example, a job that runs physics simulation needs a fast CPU but not an exceptional network connection, whereas an analysis job performing data-skimming is best located on a resource with high-bandwidth to the storage elements, but is probably not CPU-limited. By flagging workflows appropriately, the best fitting resource within all connected groups can be selected.

However, some resources available to local groups might have strict usage policies on who and what may be processed on them. Such restrictions have to be implemented on the affected resources taking advantage of the available $X.509$ infrastructure and the user's proxy. This is also only possible using mechanisms like flocking: Other concepts to integrate such resources require pilot jobs using a proxy-user, so that the remote pool administrators depend on the administrators of the submission pool to implement the correct policies.

3.2. Federating official WLCG resources

The large HEP experiments usually have access to a Grid infrastructure for performing end-user analysis, official data processing and simulation tasks. The WLCG for example is a global infrastructure, commissioned and used by multiple experiments, especially the LHC experiments. It is organized in an hierarchical structure, with the Tier 0 center located at CERN, 13 Tier 1 centers around the globe, and another roughly 160 Tier 2 centers.

This strict top-down approach requires a very well-defined setting spanning all involved centers and implies high operational costs, as each provider has to maintain particular services, provision specific environments and often pledge a 24/7 support-team. A promising alternative is shifting from dedicated, community-specific resources to inter-community shared resources to reduce the operational costs. In this setup, jobs could move freely to the most suitable resource. However, utilizing those resources, possibly located at various places, requires a change in the computing operations performed by the experiments. This is because it would not be possible for every experiment to negotiate operation models with each resource provider, even less to make the resources provide the necessary environment for HEP workflows.

In this heterogeneous environment, detailed knowledge about the resources, for instance, network or local storage, also helps to increase usage efficiency. In order to achieve this, the computing model could be adapted, such that the experiments do not decide on which particular resource to run a production job, but rather submit it to a single point of entry of a region. From there, it can further be decided whether to run the job on an associated Tier 1 or 2 resource or to send it to another federated resource, depending on where the data will be processed, or where the most suitable hardware (e.g. high throughput or GPU) is located.

4. Test setup at KIT

The recently commissioned "Throughput Optimized Analysis System" (TOpAS) at GridKa is a computing resource designed for high throughput analyses. It consists of eleven worker nodes and one management node in one HTCondor pool without interactive login. It is described in more detail in [5]. TOpAS was designed to be a shared resource for joined operation of Tier 1 and Tier 3 workflows. That is, it should run official production jobs, coming from the Tier 1 WLCG instance at GridKa in parallel with Tier 3 user jobs from the local physics community. Therefore, it is not directly integrated in the local infrastructure of the institute. As such, it is an ideal candidate for testing the federation concepts. As the next step, the two integration mechanisms flocking and using pilot jobs as discussed in [6] will be assessed for both types of workflows. After the commissioning phase it will then be integrated with the evaluated mechanisms.

The institutes HTCondor instance, used by physicists from various different experiments, is configured to flock jobs to the cluster. The authentication of hosts and users is done using the $X.509$-certificates already provided by WLCG infrastructure. As this infrastructure is already deployed, the configurational overhead is minimized and the integration is completely transparent to the users. Based upon recent positive discussions, it is planned to extend the proof-of-concept at KIT to the first external partner by end of the year.

To simplify running official Tier 1 production, as well as Tier 3 user jobs, all jobs are run in Docker containers. This way, the correct runtime environment can easily be provided, even if the operating system would not be supported by the experiment's software. All software, if not delivered with the job's payload, is provided using the "CERN Virtual Machine File System" [7] (CVMFS). This allows for lightweight container-images.

5. Summary

Using HTCondor's flocking and other mechanisms for connecting remote resource pools, a transparent integration of community-shared, distributed computing infrastructure, specialized clusters or an extension of Grid infrastructure into a single point of entry can be achieved. In combination with modern container technologies such as Docker or Singularity, this enables the usage of virtually any resource, also non-HEP dedicated resources.

With this concept, the significant amount of Tier 3 resources sitting in various institutions and accessible by their local groups can be provided to the whole community in order to maximize the overall usage efficiency.

References

[1] Thain D et al. 2005 Distributed computing in practice: the Condor experience *Concurrency and Computation: Practice and Experience* **17** 323–356 URL `https://onlinelibrary.wiley.com/doi/abs/10.1002/cpe.938`

[2] Eck C et al. 2005 *LHC computing Grid: Technical Design Report. Version 1.06 (20 Jun 2005)* Technical Design Report LCG (Geneva: CERN) URL `https://cds.cern.ch/record/840543`

[3] Docker Inc, "Docker" [software], Version 17.05.0-ce. Available from https://www.docker.com/ [accessed 2019-05-10]

[4] Kurtzer G M et al. 2017 Singularity: Scientific containers for mobility of compute *PLOS ONE* **12** 1–20 URL `https://doi.org/10.1371/journal.pone.0177459`

[5] Heidecker C et al. 2019 Boosting Performance of Data-intensive Analysis Workflows with Distributed Coordinated Caching *Journal of Physics: Conference Series (JPCS)* In this proceedings, to be published

[6] Schnepf M J et al. 2019 HEP Analyses on Dynamically Allocated Opportunistic Computing Resources *Journal of Physics: Conference Series (JPCS)* In this proceedings, to be published

[7] Blomer J et al. 2015 The Evolution of Global Scale Filesystems for Scientific Software Distribution *Computing in Science Engineering* **17** 61–71