https://doi.org/10.1051/epjconf/202024507007

Setup and commissioning of a high-throughput analysis cluster

René Caspart[1,*], *Max* Fischer[1], *Manuel* Giffels[1], *Ralf Florian* von Cube[1], *Christoph* Heidecker[1], *Eileen* Kuehn[1], *Günter* Quast[1], *Andreas* Heiss[1], and *Andreas* Petzold[1]

[1]KIT - Karlsruhe Institute of Technology, Germany

Abstract. Current and future end-user analyses and workflows in High Energy Physics demand the processing of growing amounts of data. This plays a major role when looking at the demands in the context of the High-Luminosity-LHC. In order to keep the processing time and turn-around cycles as low as possible analysis clusters optimized with respect to these demands can be used. Since hyper converged servers offer a good combination of compute power and local storage, they form the ideal basis for these clusters. In this contribution we report on the setup and commissioning of a dedicated analysis cluster setup at Karlsruhe Institute of Technology. This cluster was designed for use cases demanding high data-throughput. Based on hyper converged servers this cluster offers 500 job slots and 1 PB of local storage. Combined with the 100 Gb network connection between the servers and a 200 Gb uplink to the Tier-1 storage, the cluster can sustain a data-throughput of 1 PB per day. In addition, the local storage provided by the hyper converged worker nodes can be used as cache space. This allows employing of caching approaches on the cluster, thereby enabling a more efficient usage of the disk space. In previous contributions this concept has been shown to lead to an expected speedup of 2 to 4 compared to conventional setups.

1 Introduction

With the ever-growing amount of data needed to be processed for HEP analysis workflows, especially for future analyses at LHC Run 3 and HL-LHC, challenges for computing infrastructures arise [1]. One of these challenges is to enable workflows to run efficiently on the available infrastructures. A major factor for this is the capability to provide the required input data to the jobs. In the context of HEP workflows, this data is mainly served from storage elements located at WLCG sites. While for jobs running in the WLCG this effect can be mitigated by coordinating the jobs to run at the sites providing the data, this is not possible for end-user analysis jobs running on dedicated local facilities, for example local analysis cluster set up and commissioned at universities.

In this paper, we present an approach to designing a cluster specifically suited for running high-throughput analysis jobs, as they are common in end-user analyses in the high energy physics environment. Since this cluster is set up and commissioned close to the Tier-1 WLCG center GridKa, it is designed to profit as much as possible from this closeness, both geographically and network-topology wise, and to utilize services and infrastructure already present at

[*]e-mail: rene.caspart@kit.edu

© The Authors, published by EDP Sciences. This is an open access article distributed under the terms of the Creative Commons Attribution License 4.0 (http://creativecommons.org/licenses/by/4.0/).

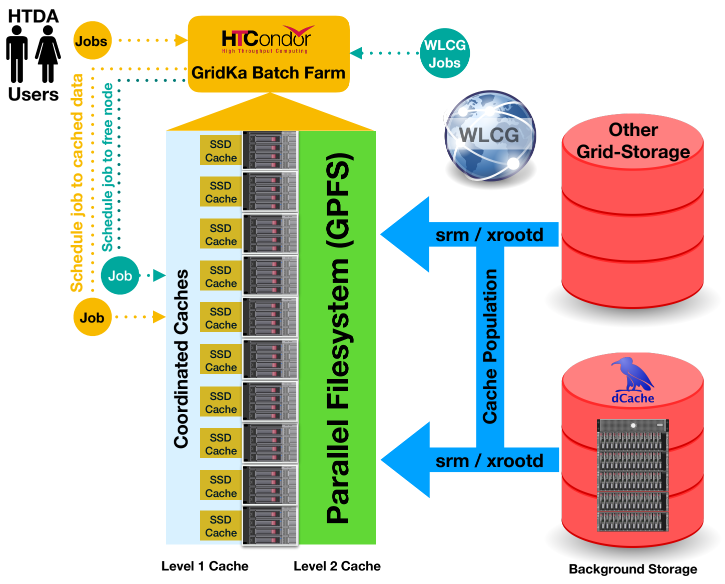

Figure 1. Design of the TOpAS cluster. The Cluster is designed to feature a hierarchical storage structure consisting of HDDs in a distributed file system and one NVMe disk for each worker node, which can be used for caching job input data.

the center. Nonetheless, the design of the cluster is chosen for it to be suitable to be deployed independently of a WLCG center at any university.

2 Design of the Cluster

The I/O performance of a cluster is one of the main bottlenecks for the efficiency of HEP jobs running on this cluster. In the design of the TOpAS cluster presented in this paper the I/O bottleneck is addressed in a twofold way. First, hyper converged worker nodes, which combine computing capabilities and storage on the same node, are chosen as the basis for the worker nodes in the cluster. These worker nodes are equipped with 16 6 TB HDD disks serving as part of a distributed storage system and one 1 TB NVMe disk serving as fast local storage device. Two 1 TB SSDs are used for the operating system and for serving local and scratch data. Each node is equipped with two Intel® Xeon® E5-2680 v4 CPUs with 14 physical cores each and 256 GB memory. This amount of memory by far surpasses the average memory requirement for HEP jobs, however it is a prerequisite for additional features considered for the cluster, such as remote direct memory access (RDMA). In addition, it enables the cluster to also be used for high-memory end-user analyses, such as machine learning tasks performed on the CPUs. In total, the cluster consists of 11 of these nodes providing up to 616 CPU threads, 2.8 TB memory and 1 PB storage on HDD drives. Second, to exclude potential bottlenecks due to network limitations, the worker nodes are connected using 100 Gbit/s interfaces. The cluster is connected to the main network backbone at the Tier-1 center GridKa with 2 100 Gbit/s connections. Consequently, access to the grid storage elements hosted at GridKa is not limited by the network connection of the cluster and data hosted at the site can be accessed with a high data-rate.

In addition to the worker nodes, a central management node is part of the cluster. This node is used for providing the services required for the operation of the cluster as well as parts of the edge-services. The design of the Cluster is illustrated in figure 1.

3 Storage Benchmarking and Performance

A key component of the TOpAS cluster is its local storage, which allows for a fast and efficient processing of jobs reading data from it. A benchmark is performed to evaluate the performance of this storage as well as identify their suitability to serve as a distributed cache for the cluster. This benchmark is designed to imitate the ideal case of a realistic analysis workflow as close as possible. The input files for the benchmarks are ROOT [2] files residing on the respective storage to be benchmarked. The data in these files is read sequentially during the benchmark. Since no further computation is performed based on the data, this benchmark serves as an upper bound for the data-rate needed by analysis workflows. The benchmark is performed using 10 worker nodes and for different numbers of processes reading the files ranging from 10, i.e. 1 per worker node, to 560, i.e. the maximum number of CPU threads for every worker node. The performance is evaluated using two metrics. For one the average throughput of the storage is evaluated. It is given by the average speed with which each process reads from the storage. The second metric is the CPU utilization for the individual processes. Assuming the process is not bound by I/O but only by the available processing power, a CPU utilization of 1.0 is expected.

The two available storage systems at the TOpAS cluster are benchmarked. For one, a CephFS [3] spanning over the HDDs of the involved worker nodes is set up. These disks are used for hosting both data and metadata of the CephFS. For the benchmarks, the CephFS is set up without enabling further mechanisms to improve the expected performance of the file system, such as Remote Direct Memory Access over Converged Ethernet (RoCE) [4]. Additionally, a benchmark is performed where the data is read from the NVMe disk of the respective worker node the process is running on. For this benchmark, the NVMe disks are set up as individual local disks on each worker node.

The results of the benchmarks can be found in figure 2. As expected the NVMe disks outperform the shared distributed file system for all number of processes due to its higher read-speed and better suitability for random I/O. While the difference in average CPU utilization for low number of processes up to 160, i.e. 16 processes per worker node, is moderate around 10%, it increases significantly for higher number of processes up to 35% at 560 processes. Even with 560 processes reading from the CephFS it is able to sustain a read rate of 25 MB/s for every process, which surpasses the average required throughput observed for most analysis workflows. In contrast to the shared distributed file system, the NVMe disks are able to sustain an almost ideal CPU utilization for the complete range of number of processes and the CPU utilization only drops for high number of processes by at most 10%.

4 Provisioning and Usage of the Cluster

To make the TOpAS cluster suitable for end-user analyses demanding high data-throughput a caching setup is used. The CephFS storage for which we reported the benchmark results in the previous section is used as cache volume, offering around 1 PB of cache space for input data. The caching is set up using the XRootD proxy file cache functionality [5]. The central management node of the cluster serves as XRootD proxy via which all XRootD file accesses on the cluster are proxied. Since the input data is only cached at the time of the first access, this setup is specifically suited for reoccurring end-user analysis workflows, where the data for subsequent accesses is then read from the cache instead of the original storage location.

For job scheduling the TOpAS cluster is set up with a dedicated HTCondor [6, 7] job scheduler. This scheduler is chosen since it is commonly used in the HEP environment and has already successfully been employed both at the Tier-1 WLCG center GridKa and for the local resources at the institute for experimental particle physics in Karlsruhe. Among others,

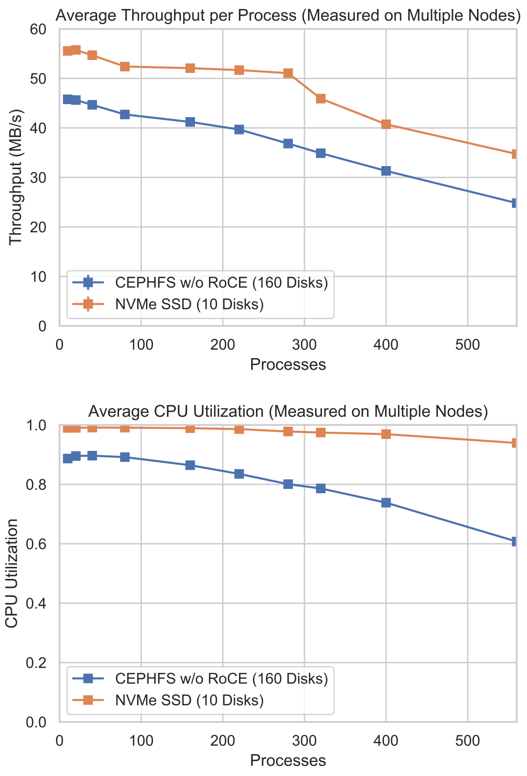

Figure 2. Results of the reading benchmark performed on the TOpAS worker nodes. The average throughput of the storage per process accessing it is shown at the top and the average CPU utilization for the processes at the bottoms.

a key reason for choosing the HTCondor scheduler was the possibility to easily enable jobs from users submitted on the local resources at the institute to run on the TOpAS cluster. This is achieved using a mechanism provided by HTCondor for connecting multiple HTCondor resource pools, called *flocking* [8]. Details of this setup have been reported in [9]. The usage of this dedicated mechanism is necessary for two reasons. First, the local resources and servers with enabled user login at the institute and the TOpAS cluster are hosted in an independent network infrastructure. Second, the TOpAS cluster and local resources employ independent user bases. While the local resources offer individual accounts for each user at the institute the TOpAS cluster is only set up with few commonly used pool-accounts. This and the fact that no dedicated login nodes are provided requires that interaction of the users with the TOpAS cluster is limited to the batch system scheduler as an entry point.

As a result of performance tests for different number of threads used per worker node, each worker node in the cluster is set up to provide 42 CPU threads in the batch system, leading to a total of 462 CPU threads being available. End-user analysis jobs on the TOpAS cluster are executed in a docker container [10]. This isolates them from the system, while enabling users to choose from a wider range of operating systems, such as Scientific Linux 6 and 7, and use containers already providing parts of the required software stacks. In addition, this setup mirrors the setup used for the local institute resources, thereby limiting the need for users to adapt their workflows. An example of a typical workload of end-user analyses in a week on the TOpAS cluster is shown in figure 3. It can be seen, that the submission of jobs running on the TOpAS cluster happens in burst-like patterns and at times no end-user

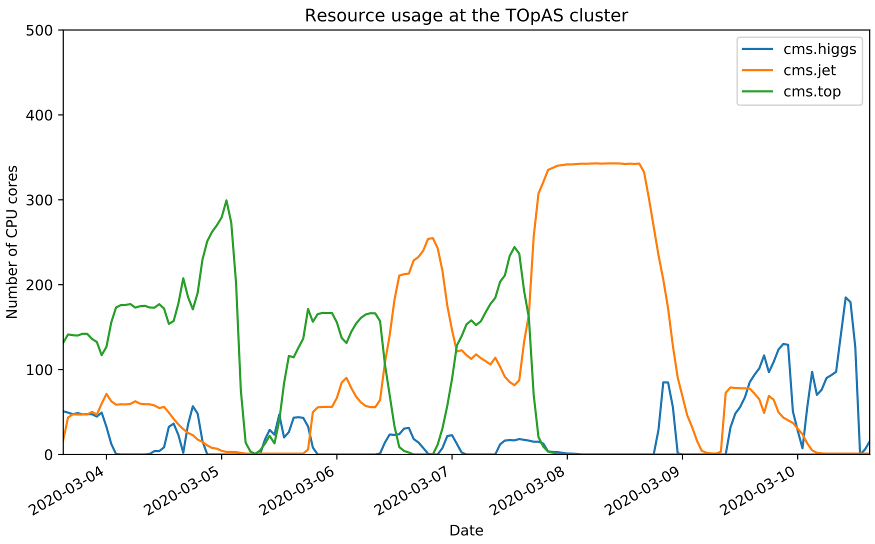

Figure 3. Utilization of the TOpAS cluster by jobs submitted from end-users at the institute for experimental particle physics in a seven-day period. Workloads typically occur in bursts, leading to times, where the resources are not fully utilized for end-user analyses. The workloads are shown split by accounting group, which corresponds to the field of research the users are involved in.

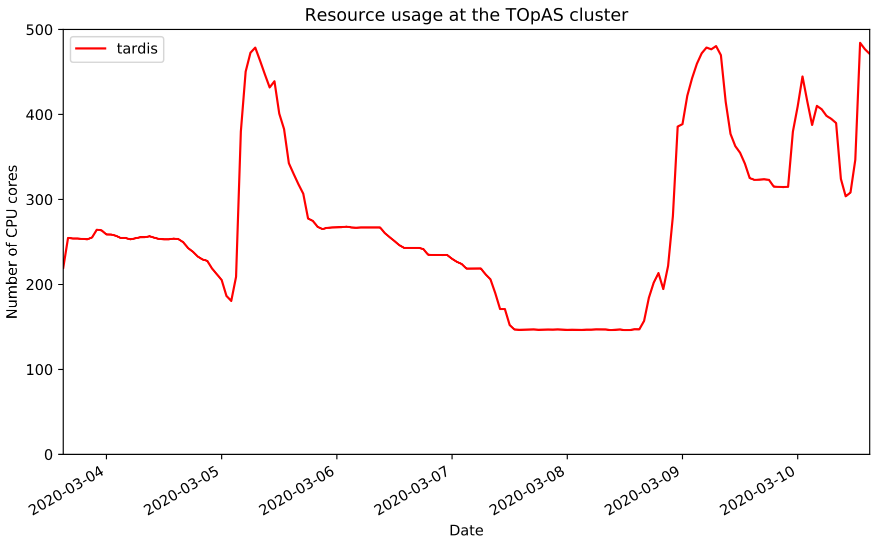

Figure 4. Utilization of the TOpAS cluster for WLCG workloads submitted by the CMS and Belle II collaborations in a seven-day period. The workloads are run when the cluster is not fully utilized by end-user analysis workflows.

jobs are running on the cluster at all. Considering only end-user analyses as a use case for the cluster, this would lead to the cluster not always being fully utilized and resources idling.

At times at which the cluster is not fully utilized by end-user analyses otherwise idling resources are used for back filling with WLCG workflows by the CMS and Belle II collaborations. Since the job submission by the users shows no clear structures and is hard to predict [11] this information can not be used for scheduling jobs by the collaborations on the TOpAS cluster in times the cluster is idle. Instead, the workloads are started in preemptable slots, enabling vacating resources in favor of end-user analyses whenever needed. For setting up the back filling the resource manager COBalD [12] and TARDIS [13], which are developed at Karlsruhe Institute of Technology (KIT), are used. A dedicated infrastructure hosted at the WLCG Tier-1 center GridKa is used for accepting and scheduling jobs to opportunistic resources, such as the resources used for back filling at the TOpAS cluster. Details on the resource manager COBalD and TARDIS as well as their setup and deployment and the infrastructure at GridKa can be found in [14, 15]. Backfilling jobs are run inside singularity container [16] to offer the same environment as expected by the collaborations for WLCG worker nodes without the need for deploying the required software on the TOpAS worker nodes directly. The backfilling jobs are allowed to run on the TOpAS cluster indefinitely as long as no suitable user jobs are available. In case suitable user jobs are available, the backfilling jobs are preempted and resources are freed with a short grace period. This setup is chosen to ensure the high-throughput resources are available for processing user jobs with a minimal waiting time, while still enabling the back filling jobs time to reach a state, where the loss of computing time due to the preemption of the job is minimal. At the time of writing over 90,000 CPU core-hours at the TOpAS cluster were provided opportunistically to the CMS collaboration.

5 Conclusion

In this paper, we presented the design and commissioning of a cluster specifically suited for high-throughput analysis workflows. This cluster was set up at Karlsruhe Institute of Technology close to the Tier-1 WLCG computing center GridKa. It was designed to profit from already existing infrastructure of the WLCG site while also being flexible enough to be suited as a prototype fit for deployment at smaller sites and universities. The cluster is based on hyper converged worker nodes providing both compute and storage capabilities. We presented benchmark results for the storage of the cluster, which is used as a cache for data accessed via XRootD by jobs running on the cluster. These benchmarks illustrate the capabilities of these storages to serve data for data-intensive workflows running on the cluster and show their suitability for concurrent accesses by jobs running on the cluster.

While the cluster primarily is designed and used for data-intensive end-user analysis workflows, submitted by users at KIT, we reported on the possibility and employed setup to successfully use idling resources during times of little job pressure for back filling with WLCG jobs by the CMS and Belle II collaborations.

In addition to the setup presented in this paper, future work will focus on improving the gain from this setup. Firstly, the choice of the distributed file system used for caching can be revisited. This includes considerations like enabling RoCE for the CephFS, but also considerations for alternative file systems such as IBM spectrum scale [17], which is a commonly used choice in computing centers such as GridKa. Secondly, for the caching setup two improvements can be followed up on. First, the design offers the possibility for a cascaded caching approach. In this approach, both the CephFS and NVMe disks will be used for caching, with the more performant NVMe disks serving as local level 1 cache for each worker node and the CephFS serving as larger distributed level 2 cache. Second, mechanisms can be studied to improve the coordination of jobs requesting input data to the TOpAS cluster and in the case of local caches to the worker node where a file is cached. Plans for this approach will be based on the coordinated distributed caching approach studied at KIT in the past and reported in [18, 19].

Acknowledgment

The authors acknowledge support by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) through the project "Massenspeicher und Rechenknoten für hohen Datendurchsatz", project number 398475463.

References

[1] HEP Software Foundation, *A Roadmap for HEP Software and Computing R&D for the 2020s*, Computing and Software for Big Science 3, 7 (2019), DOI: 10.1007/s41781-018-0018-8

[2] Rene Brun and Fons Rademakers, *ROOT - An Object Oriented Data Analysis Framework*, Nucl. Inst. & Meth. in Phys. Res. A 389 (1997) 81-86. See also http://root.cern.ch/

[3] Sage Weil and Scott Brandt and Ethan Miller and Darrell Long and Carlos Maltzahn, *Ceph: A Scalable, High-Performance Distributed File System*, OSDI 2006-11, pp 307-320

[4] *InfiniBand^{TM}Architecture Specification Release 1.2.1 Annex A17: RoCEv2*, InfiniBand Trade Association, 2 September 2014, https://cw.infinibandta.org/document/dl/7781

[5] L. A. T. Bauerdick et al., *XRootd, disk-based, caching proxy for optimization of data access, data placement and data replication*, Journal of Physics: Conference Series513(2014) 042044, DOI: 10.1088/1742-6596/513/4/042044

[6] Todd Tannenbaum, Derek Wright, Karen Miller, and Miron Livny, *Condor - A Distributed Job Scheduler*, Beowulf Cluster Computing with Linux, The MIT Press, 2002. ISBN: 0-262-69274-0

[7] HTCondor Team, *HTCondor*[software], DOI: 10.5281/zenodo.3595387

[8] D. H. J Epema, Miron Livny, R. van Dantzig, X. Evers, and Jim Pruyne, *A Worldwide Flock of Condors : Load Sharing among Workstation Clusters*, Journal on Future Generations of Computer Systems, Volume 12, 1996. DOI: 10.1016/0167-739X(95)00035-Q

[9] Ralf Florian von Cube et al., *Federation of compute resources available to the German CMS community*, Journal of Physics: Conference Series ACAT 2019 proceedings (to be published),

[10] Docker [software], https://www.docker.com/ [accessed 2020-03-01]

[11] Eileen Kuehn, et al., *Predicting resource usage for enhanced job scheduling for opportunistic resources in HEP*, EPJ Web of Conferences CHEP 2019 proceedings (to be published)

[12] Max Fischer, Eileen Kuehn et al., *COBalD - the Opportunistic Balancing Daemon*, matterminers/cobald [software], DOI: 10.5281/zenodo.3469929

[13] Manuel Giffels, Matthias Schnepf et al., *TARDIS Resourcemanager*, matterminers/tardis [software], DOI: 10.5281/zenodo.3688615

[14] Max Fischer et al., *Lightweight dynamic integration of opportunistic resources*, EPJ Web of Conferences CHEP 2019 proceedings (to be published)

[15] Manuel Giffels et al., *Effective Dynamic Integration and Utilization of Heterogenous Compute Resources*, EPJ Web of Conferences CHEP 2019 proceedings (to be published)

[16] Kurtzer GM, Sochat V, Bauer MW (2017) *Singularity: Scientific containers for mobility of compute*, PLoS ONE 12(5): e0177459, DOI: 10.1371/journal.pone.0177459

[17] Dino Quintero et al., *IBM Spectrum Scale (formerly GPFS)*, IBM Redbook SG24-8254-00, ISBN: 0738440736

[18] Max Fischer et al., *Opportunistic data locality for end user dataanalysis*, Journal of Physics: Conf. Series 898 (2017) 052034, DOI: 10.1088/1742-6596/898/5/052034

[19] Christoph Heidecker et al., *Advancing throughput of HEP analysis work-flows using caching concepts*, EPJ Web Conf. 2019-214, DOI: 10.1051/epjconf/201921404007