EPJ Web of Conferences **173**, 05010 (2018) https://doi.org/10.1051/epjconf/201817305010
Mathematical Modeling and Computational Physics 2017

Application of SLURM, BOINC, and GlusterFS as Software System for Sustainable Modeling and Data Analytics

Vladislav V. Kashansky[1,⋆] and *Igor L.* Kaftannikov[1,⋆⋆]

[1] *School of Electrical Engineering and Computer Science, South Ural State University, Russia*

Abstract.
Modern numerical modeling experiments and data analytics problems in various fields of science and technology reveal a wide variety of serious requirements for distributed computing systems. Many scientific computing projects sometimes exceed the available resource pool limits, requiring extra scalability and sustainability. In this paper we share the experience and findings of our own on combining the power of SLURM, BOINC and GlusterFS as software system for scientific computing. Especially, we suggest a complete architecture and highlight important aspects of systems integration.

1 Introduction

Distributed environments have become very influential in the contemporary data processing and analytics. During several decades, we had been observing architecture changes emerging to support computation trends of various forms. Among them, there were hardware-defined platforms, SSI-based systems [1] and clustered multi-node solutions, orchestrated by different batch schedulers. Contemporary computing architectures [2] involve clustering approach, managing computing processes among the nodes of a large distributed computing platform. Many papers have been written about the general nature of the distributed computing. In this paper, we are going to discuss and focus on more "local" and application-specific tasks.

The problem of infrastructure deployment arose during 2015–2016, when we decided to process medium-sized data sets and perform corresponding computations for several projects led by Dr. I.L. Kaftannikov. At early stages, we were using volunteer computing resources managed by the BOINC system. The more data we wanted to process, the more inner resources we had to involve. The problem is that with the growth of the project, we had to redefine the local infrastructure, making it more reliable and stable. We also had some computational jobs, which required additional control and it was no longer possible to use it in the context of BOINC, as we required lower latencies and higher degree of security. At the same time, we wanted to save the existing approach, allowing volunteers to integrate their computational resource base.

⋆e-mail: vladislav.kash@gmail.com
⋆⋆e-mail: kil7491@gmail.com

2 Early Stages Architecture. BOINC System

As we stated above, we decided to use the BOINC system as initial solution. It is a free software platform [3] developed by the Berkeley University, implementing a volunteer public-resource model of distributed computations. BOINC allowed us to involve different types of computing powers, connecting many individual computers to a common computational network. Owners of physical or virtual computing resources were able to join us publicly via the BOINC client program and to help by contributing some resources.

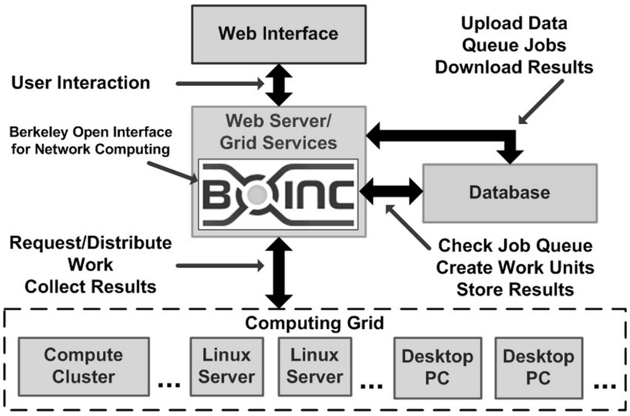

Figure 1. Architecture of the BOINC project. The platform is centralized and includes server and user-side parts. User-side part consists of a client and a graphical program – BOINC Manager. The manager brings the GUI for controlling of one or many BOINC clients. Proper details of BOINC project can be found in [3], improvements in [4]. The most recent changes and documentation can be obtained from the Github repository and the official website.

Many popular BOINC projects [3] demonstrate significant computational performance, but with some strict limitations and only over a certain class of jobs. This of course allows scientific groups to carry out resource-intensive calculations without the use of expensive hardware. But with this approach we faced two major architectural problems:

1. The architecture of data storage is undefined as the BOINC system operates at a higher layer of data abstraction.
2. The scheduling model within BOINC system is inconsistent with the local jobs distribution, which asks for a fast acting system with low transition rates.

Based on Refs. [5–7] we decided to use the SLURM as a local batch scheduler and GlusterFS as a distributed file system for storing experimental data sets.

3 Redefined Architecture. GlusterFS and SLURM

GlusterFS is a distributed, parallel, linearly scalable, failure-tolerant file system. We used it in the context of TCP/IP networking to export data "bricks" from different servers, forming a single parallel distributed data space. GlusterFS operates in user space, based on FUSE technology and works on the top of the existing file systems, so we didn't have to face the kernel support problem. Unlike other distributed file systems, such as Lustre and Ceph [5], GlusterFS does not require a separate server for storing metadata[1], providing a great degree of flexibility and scalability. By bringing GlusterFS to our BOINC stack we changed the storage mechanisms, allowing storing our data in a more efficient way.

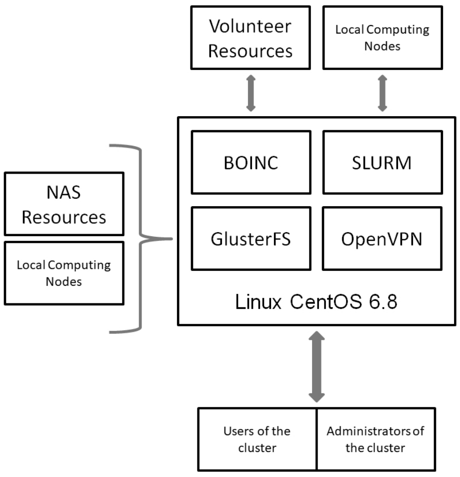

Figure 2. Diagram of the new architecture. The whole solution runs on CentOS 6.8. BOINC and SLURM form computational core. Custom meta-scheduler controls the job distribution process. It classifies jobs and sends them into the correct queue according to the specific scientific workflow. GlusterFS provides extensible single parallel distributed data space. OpenVPN forms a secure overlay network for communication of the inner services. We used it to connect several storage nodes that weren't accessible directly via IPv4 address. Administrators and users interact with the system via SSH commands and GUI.

The other valuable part of our new architecture is SLURM. The developers describe it as scheduling system for large and small Linux clusters. It requires no kernel modifications for its operation and is relatively self-contained [7]. However, the integration with SLURM required jobs classification which was done within the following scheme:

1. Jobs with low data exchange intensity, possibility of execution pausing and reassignment[2] are running on the volunteer resources.

2. Jobs with intensive data exchange or strict deadlines are running on the local cluster and scheduled via SLURM.

After performing the integration depicted in figure 2, SLURM filled an important gap in our architecture, providing the ability for starting, executing, and monitoring jobs on the set of locally allocated nodes. Now, when a user registers the specific job in system's workflow, meta-scheduler can run it locally, or assign it to any possible volunteer's computing resource, depending on the set of preferences. The distribution scheme [8–10] as well as whole infrastructure tuning, monitoring [11] and maintenance can turn to really complex solution. This is beyond the scope of this work, but we suggest papers and books mentioned in our references for interested readers.

[1]GlusterFS uses Elastic Hashing Algorithm (EHA).
[2]Volunteers are usually able to pause or even cancel assigned jobs, making execution time unpredictable.

4 Conclusion

The use of Distributed File Systems and Batch Schedulers is a common practice for clusters, datacenters, and supercomputers. In this paper, we tried to highlight a radical infrastructure redefinition, joining high-latency loosely-coupled volunteer public resources with local high-performance cluster. The use of SLURM, BOINC and GlusterFS as a platform for our computations allowed us:

1. To get a horizontally scalable computing system with minimal costs for organization and subsequent maintenance. We used the available local and volunteer computing resources and spent around 1200 man-hours on planning and development.
2. To solve the resource idleness problem. We connected 20 Intel Core I7-4770 2x4096DDR3 SATA 1TB via BOINC and 10 Intel Xeon-X5680 1x4096DDR3 SAS 1TB nodes via SLURM.
3. To socialize research of the department, allowing outside participants offer their own resources. Volunteers donated more than 200 heterogeneous nodes, including several server farms.
4. To process wider spectrum of jobs due to combining local high-performance cluster with BOINC system.
5. To automate functions related to the jobs distribution, results aggregation, and reporting data creation via a meta-scheduler and a set of custom scripts.

References

[1] L. Renaud, P. Gallard, G. Vallée, Ch. Morin, and B. Boissinot, *IEEE International Symposium on Cluster Computing and the Grid (CCGrid 2005)* **2** (Cardiff, Wales, UK, 2005) pp. 1016–1023

[2] S. H. Fuller and L. I. Millett, *The Future of Computing Performance: Game Over or Next Level?* (National Academy Press, Washington DC USA, 2011)

[3] D. P. Anderson, *Fifth IEEE/ACM International Workshop on Grid Computing* (Pittsburgh PA USA, 2004) pp. 4–10

[4] V. V. Kashansky and I. L. Kaftannikov, *Selected Papers of the 7th International Conference Distributed Computing and Grid-technologies in Science and Education* (Dubna, Russia, 2016) pp. 284–288

[5] B. Depardon, G. Le Mahec, and C. Séguin, *Analysis of Six Distributed File Systems* (Research Report, The open archive HAL, 2013)

[6] A. Davies and A. Orsaria, Linux Journal **2013** (235), 72–82 (2013)

[7] A. B. Yoo, A. J. Morris, and M. Grondona, *Job Scheduling Strategies for Parallel Processing* (Springer, Berlin, Heidelberg, 2003) pp. 44–60

[8] V. Berten, J. Goossens, and E. Jeannot, IEEE Transactions on Parallel and Distributed Systems **17** (2), 275–278 (2006)

[9] N. A. Balashov, A. V. Baranov, I. S. Kadochnikov, V. V. Korenkov, N. A. Kutovskiy, A. V. Nechaevskiy, and I. S. Pelevanyuk, *Selected Papers of the 7th International Conference Distributed Computing and Grid-technologies in Science and Education* (Dubna, Russia, 2016) pp. 114–118

[10] M. Combarro, A. Tchernykh, D. Kliazovich, A. Drozdov, and G. Radchenko, *2016 International Conference on Engineering and Telecommunication (EnT)* (Moscow, Russia, 2016) pp. 29–33

[11] I. S. Kadochnikov and I. S. Pelevanyuk, *Selected Papers of the 7th International Conference Distributed Computing and Grid-technologies in Science and Education* (Dubna, Russia, 2016) pp. 275–278