Embedding Cloud Computing inside Supercomputer Architectures

Patrick Dreher and Mladen Vouk
Department of Computer Science, North Carolina State University, P.O. Box 8206, Raleigh, North Carolina, U.S.A.

Keywords: High Performance Computing, Cloud Computing, Scientific Workflows, Virtual Computing Laboratory, Supercomputing, HPC/Cloud, Supercomputer/Cloud, Future Supercomputer/Cloud Architectures.

Abstract: Recently there has been a surge of interest in several prototype software systems that can embed a cloud computing image with user applications into a supercomputer's hardware architecture. This position paper will summarize these efforts and comment on the advantages of each design and will also discuss some of the challenges that one faces with such software systems. This paper takes the position that specific types of user applications may favor one type of design over another. Different designs may have potential advantages for specific user applications and each design also brings a considerable cost to assure operability and overall computer security. A "one size fits all design" for a cost effective and portable solution for Supercomputer/cloud delivery is far from being a solved problem. Additional research and development should continue exploring various design approaches. In the end several different types of supercomputer/cloud implementations may be needed to optimally satisfy the complexity and diversity of user needs, requirements and security concerns. The authors also recommend that the community recognize a distinction when discussing cluster-type HPC/Cloud versus Supercomputer/Cloud implementations because of the substantive differences between these systems.

1 INTRODUCTION

Over the past few years cloud computing has rapidly gained acceptance as both a working technology and a cost effective business paradigm. When applied to certain user applications, these advances in hardware and software architectures and environments are now able to deliver reliable, available, serviceable and maintainable cloud systems that are cost effective.

The success of these cloud systems has encouraged designs that extend these platforms to high performance computing applications. Today commercial Cloud platforms such as Amazon Web Services (Amazon), and a number of others, offer clouds platforms for HPC applications.

Despite all of these advances, cloud computing has only had mixed success servicing high performance computing (HPC) applications (Parashar et al.), (Yelick et al.). Initial attempts to migrate these applications to cloud platforms did show promise in cases of minimal inter-processor communication, but more tightly coupled HPC applications suffer degraded performance. Most cloud computing systems lack the specialized HPC architectural network infrastructure needed to satisfy the minimum latency requirements for these codes. Various custom built HPC/cloud systems (Vouk, 2008), (Vouk, et. al., 2009), (Vouk, et. al, 2010) and commercial platforms such as Penguin Computing (Penguin) and others added the needed high speed network interconnects to provide the cluster hardware with the network communications for low latency HPC applications.

Although these designs helped, in general large tightly coupled HPC applications require hundreds to thousands of compute nodes. This is problematic for rack based clusters because it is quite likely that such an assembled hardware configuration will contain motherboards that lack uniformity in the chipsets and clock frequencies. For HPC applications that depend on tightly coupled hardware infrastructures, these incommensurate CPU clock frequencies, network shortcomings and other in homogeneities in the overall hardware system discount cloud computing clusters as an option for tightly coupled HPC applications.

Other options that have been explored for running HPC applications in clouds include constructing small groups of HPC clouds with more robust uniform hardware architectures and network connections. These systems are configured for spill-over provisioning (bursting) from the HPC supercomputer to these cloud systems when the supercomputer becomes saturated. Although these alternatives provide some overall acceleration, the underlying shortcomings of delivering supercomputer level computational throughput with commodity cloud cluster hardware still remains problematic.

In addition to the hardware architecture requirements, HPC users with tightly coupled applications want systems where they can operate them “close to the metal”. This includes the ability to tune hardware, software and storage in order to optimize computational performance and throughput. Adding to these requirements are the increasing amounts of output data from applications or ingestion of large quantities of data from sensors, laboratory equipment, or data on existing storage systems. This I/O need has added additional complications to these design requirements. Ideally, many HPC users are also interested in creating workflows that allow seamless transitions between computation and data analysis, preferably all managed from the user’s desktop.

HPC data centers are also showing increased interest in the idea of a supercomputer/cloud option. HPC facilities are seeing rapid expansion in both the number of users and types of applications at these centers. Today there is a greater dispersion in the level of user expertise when compared to decades ago. Researchers who have been using Supercomputer Centers for decades are usually well versed in the technical complexities and expertise required to utilize these systems. These collaborations have experts with the technical knowledge and breadth of experience to handle the challenges and complexities of porting and tuning the collaboration’s specific systems to each HPC platform. However, many other user groups are either new to the world of HPC or the size of the group is relatively small. In these cases, the internal overhead required by the collaboration to address these porting and tuning issues may be daunting to the point where it is stunting progress for these research groups. Adding to this complexity, today many research communities are using computational and data analysis software environments compatible with their experimental equipment or data systems but which may not easily be installed in an HPC center. Servicing these requirements among the different research communities is becoming both technically and financially challenging.

Today computational hardware architectures and software environments have now advanced to the point where it is possible to consider building a supercomputer/cloud platform as a viable option for tightly coupled HPC applications and projects with specialized software environments. This position paper is organized as follows. Section I outlines some of the challenges for using clouds for HPC applications and the considerations and some potential advantages if such systems could be built. Section 2 outlines some of the characteristics needed for a supercomputer/cloud system. Section 3 discusses some of the prototype designs for implementing a supercomputer/cloud system and the potential constraints and limitations that such systems may encounter. Section 4 advocates the position that various types of supercomputer/cloud systems should be constructed to properly support tightly coupled HPC applications needing the large computational platforms and flexible software environments.

2 SUPERCOMPUTER/CLOUD SYSTEM CHARACTERISTICS

The goal of a supercomputer/cloud system is to enable a user application to run in a secure cloud framework embedded directly inside a supercomputer platform in such a way that the cloud can capitalize on the HPC’s hardware architecture. This requires carefully combining of components from both supercomputer and cloud platforms.

The supercomputer provides the low latency interconnects between processors and allows the cloud image to utilize the homogeneous and uniform HPC system-wide computational hardware. HPC applications that typically run on supercomputers are parallelized and optimized to take maximum advantage of the supercomputer’s hardware architecture and network infrastructure. These applications also utilize custom software libraries tuned to the specific HPC architecture, thereby achieving excellent high performance computing throughput. However, users cannot elastically provision these supercomputer systems.

A cloud system on the other hand, does allow a user to request, build, save, modify and run virtual computing environments and applications that are reusable, sustainable, scalable and customizable. This elastic resource provisioning capability and the multi-tenancy option are some of the key underpinnings that permit architecture independent scale-out of these resources. Although this provides the users with elasticity, this type of cloud system design allows for little or no direct hardware access, restricts flexibility for specialized implementations, and limits the overall predictability and performance of the system.

A supercomputer/cloud system provides users with flexibility to build customized software stacks within the HPC platform. These customizations may include implementation of new schedulers and configuration of operating systems that may be different from the native OS and schedulers on the supercomputer platform itself. This type of flexibility within the same supercomputer hardware platform can support both computation and data analysis using the supercomputer and storage platform where the data physically resides, thereby alleviating the need to move data from one physical location to another. Finally, an HPC hardware architecture with the elasticity of a cloud computing system may have economic and cost savings. A supercomputer platform that can serve as both an HPC and cloud system may provide economies of scale for the Data Center.

3 PROTOTYPES DESIGNS FOR SUPERCOMPUTER/CLOUD SYSTEMS

Projects are underway today that are exploring the technologies for embedding a cloud computing capabilities within a supercomputer hardware platform. Within the last few years several prototypes have emerged for building and integrating cloud systems and the host's supercomputer hardware platform. They include both embedding a full cloud image within a supercomputer and container designs that ride on top of an OS supporting that cloud application.

3.1 Full OS Kernel Cloud Image

Initial attempts to design an infrastructure capable of hosting a full cloud image inside a supercomputer were pioneered by IBM Research (Appavoo, 2009). The team at IBM focused on the idea of developing software that can access and capitalize on the network infrastructure of a Blue Gene/P (BG/P) supercomputer as a host machine for supporting a public utility cloud computing system. The IBM Group developed an open source software utility toolkit (Appavoo, 2008), (Appavoo, 2012) built around a group of basic low level computing services within a supercomputer.

This design included defining four basic building blocks of owners, nodes, communications domains and control channels. By definition, the user implementing the toolkit is the default owner of this process. The nodes identified within the BG/P that are accessed by the Kittyhawk utility provide a flat uniform physical communication name space and are assigned a unique physical identity. Each node was configured with a control interface that provided the owning process access to the node via an encrypted channel. Nodes were always able to establish a communications channel with each other by sending messages using the physical identification property within the machine to locate every other node. The control channel provided the access interface between the owning process and the allocated nodes within the supercomputer. The node control channel also provided a user with a mechanism to access the raw hardware. This design enabled customized allocation and interconnection of computing resources and permitted additional higher levels of applications and services to be installed using standard open-source software.

Applying the ideas from the IBM Group's work, a prototype Infrastructure as a Service (IaaS) cloud computing system was constructed inside an IBM Blue Gene/P supercomputer (Dreher 2014). The cloud computing system selected for porting to this supercomputer/cloud software architecture was the Virtual Computing Laboratory (Apache VCL, 2016). This cloud system was a thoroughly tested open source software implementation that has been operational and in production since 2003.

The VCL front end managed requests and scheduling for supercomputer/cloud jobs. The VCL login communicates this information to the BG/P login node that is listening for cloud computing service requests. The login node establishes communications and control channels to the designated management node within the cloud environment within the BG/P itself. Information is passed from the BG/P login node to the management node established inside the BG/P and worker nodes are allocated to run the cloud session inside the BG/P supercomputer.

The original ideas of Appavoo, et. al. and the IaaS prototype implementations within an IBM Blue Gene/P by Dreher and Georgy have been extended by other groups (AbdelBaky, et. al.) also using a BG/P supercomputer platform. Working with colleagues at IBM's T.J. Watson Research Center a software utility resource manager (Deep Cloud) was developed that can handle interactive workloads or those requiring elastic scaling of resources for supercomputer/cloud IaaS instances (Shae, et.al.). To extend the supercomputer/cloud for PaaS a software utility called "Comet Cloud" (Kim and Parashar) (Comet Cloud) was developed. This software has four building blocks (server, master, worker, and Comet space). These components essentially control and manage the cloud request and steer the job through the supercomputer platform. This includes providing load balancing, including scheduling and mapping of all application tasks, provisioning of resources, workflow and application and system level fault tolerance.

The advantages of these prototypes are that users can tap the supercomputer's hardware infrastructure and run the application on a dedicated set of nodes that are specifically designed for processing HPC applications. The cloud images used for these implementations can be constructed from a library of cloud images. The user also has the other option of running virtualization software on a portion of the supercomputer within the allocated communications domain. The disadvantages of these supercomputer/cloud designs is that the utility toolkits may be platform specific and that a new customized toolkit would need to be implemented for other supercomputer platforms (Dreher, 2015).

3.2 Containers

An alternative approach to building supercomputer/cloud platforms is to utilize the concept of container based virtualization (Soltesz, 2007). The basic idea is to construct a framework that provides a comprehensive abstraction layer and suite of management tools (Poettering, 2012), (Graber, et.al.), (Marmol), Cloud Foundry Warden) and (Solomon) capable of running multiple isolated Linux Systems under a common host operating system. This type of design is focused toward a capability of allowing the container application to run on a variety of computational hardware infrastructures.

A container based system design deployed inside a supercomputer platform depends on the idea that only a single copy of the kernel needs to be installed on this system. The container itself is just a process that operates on top of that single installed kernel. As a result, a container usually requires significantly less memory that a full Virtual Machine (VM) because it is only running a specific application or process at a given time.

This container design offers operational efficiencies when compared to the software layers needed to build the full VM implementation. Because a container essentially represents only a process, it does not carry the full overhead of initializing a virtual machine and booting an entire kernel at start-up for each instance. This streamlining of the kernel can improve installation times from several minutes for full VM install down to a few seconds for a light-weight container. In addition, because of the reduced kernel size, the memory requirements for the installation are considerably reduced when compared to a full VM implementation. The reduced OS does not have the overhead and burden of additional software layers and may be easier and more manageable for certain users and applications that do not require this more customized tuning.

File system access is also a topic that has been addressed with a container design. For the full VM install option each VM must run an instance of the file system client. This can become problematic and increase the overhead if the number of VM instances operational at any given time is large. The container model will utilize the file system from the host and map it to each individual container, thereby leaving only one instance of the file system client.

These advantages are extremely attractive to supercomputer/cloud designers. Just as with the full cloud image supercomputer/cloud designs, containers also offer a mechanism for providing customized software stacks to individual scientific communities and project collaborations using supercomputer data centers. Unlike the full cloud image installs, containers can be activated in a fraction of the time compared to a full cloud image installation. The single host operating system can even support individual modules of a software stack at any given time. Less technically sophisticated HPC users or small collaborations without the breadth of dedicated technical expertise within their groups to support the complex process of code porting to HPC platforms can utilize container technology to move their research forward.

The HPC Data Centers are also hoping that some type of design along these lines can better support the growing number and variety of different types of users. Exploring these ideas, various HPC Data Centers have begun to experiment with supercomputer/cloud container prototype systems. Jacobsen and Canon (Jacobsen and Canon, 2015) have moved forward with a prototype installation of Docker container at NERSC (NERSC). Users first need to either select or create a Docker image and then move it to a DockerHub using the NERSC custom designed software system called Shifter. The shifter software prepares the container. It also modifies the image to prevent users from running processes with elevated privileges beyond the user level as well as other steps to prevent security risks. When the Shifter process is complete the Docker image is submitted as a batch job. This container based approach to an HPC/Cloud has been extended by Higgins Holmes and Colin (Higgins, Holmes, Colin, 2015) who tested MPI codes running inside Docker containers in HPC environments. What these authors found was that there was little degradation in performance when compared to running the MPI code directly on the HPC system. This result offers the potential to expand an HPC/Cloud capability via a container implementation to a large number of HPC applications.

4 POSITION SUMMARY

The authors would also like to suggest that the community adopt a distinction when discussing HPC/Cloud implementations versus supercomputer/cloud implementations. Although a cloud running HPC applications on a cluster type configuration may sometimes deliver somewhat comparable levels of performance when compared to a supercomputer platform, there are inevitable scaling issues, constraints and operational costs between these two hardware architectures and software environments.

The examples cited in this paper suggest that both the full cloud image and container approaches show considerable promise for future supercomputer/cloud production systems. However, within the supercomputer/cloud context, different designs and options may need to be developed to address the diverse set of HPC user needs and requirements. For example, a container implementation can offer fast start-up and response times for highly dynamic workloads that require rapid on-demand shifting of resources within the system. Users applying complex workflows and experimentation with new schedulers and operating systems within a working HPC environment may prefer a supercomputer/cloud environment with a full cloud image and kernel implementation for each installed instance with the supercomputer.

However enticing all of this seems, this paper urges caution. Although each of these supercomputer/cloud implementations has specific advantages, these designs also raise serious challenges in customization, operations and computer security. Both an unmodified full cloud image and a container have the potential to run on the supercomputer platform with elevated privileges beyond the user level if submitted without modifications. In the case of containers, with only one Linux kernel servicing multiple processes, it is possible for a user running in one process to disrupt other user processes in separate containers. The Linux system does not offer strong isolation for I/O operations and that can lead to other potential difficulties.

The costs of mitigating these challenges pose downstream financial and operational concerns for HPC providers. Data Centers implementing either a full cloud image or containers generally must make some configuration adjustments before these supercomputer/cloud prototypes can securely run on their supercomputer platforms. In addition, migration of these supercomputer/cloud prototypes to other HPC platforms will likely require additional customizations specific to each new system, with cost implications both for developers and for the data centers operations groups. However, if these issues can be solved, the data center operational costs of supporting one overall hardware platform that can deliver supercomputer, cloud and data analytics may be substantial.

The exuberant embrace of any one prototype as the supercomputer/cloud solution is discouraged at this point. Different types of applications may tend to favor one design over another and at this point it is too early to declare that any one design approach satisfies all supercomputer/cloud requirements. There is still much work to do with the experimentation and testing of supercomputer/cloud system designs against HPC applications may take the form of tightly coupled, loosely coupled or simple disconnected parallel implementations. In the end, there may need to be both a full cloud image and container supercomputer/cloud implementations to optimally address the complexity and diversity of the needs, requirements and computer security concerns of this ever growing and expanding HPC user community. Based on the information provided, we hope that our position paper moves the community to adapt and label a supercomputer/cloud as a distinctly separate design from an HPC/Cloud cluster hardware system and move forward recognizing this distinction as these systems evolve.

ACKNOWLEDGEMENTS

This work was supported in part through NSF grants 0910767, 1318564, 1330553, the U.S. Army

Research Office (ARO) grant W911NF-08-1-0105, the Science of Security Lablet, the IBM Share University Research and Fellowships program, the Argonne Leadership Computing Facility at Argonne National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under con- tract DE-AC02-06CH11357, the NC State Data Science Initiative, and a UNC GA ROI grant. One of us (Patrick Dreher) gratefully acknowledges support with an IBM Faculty award.

REFERENCES

AbdelBaky, et. al. "*Enabling High-Performance Computing as a Service*", Computer, Issue No.10, vol 45, (Oct. (2012) pp 72-80.

Amazon Web Services https://aws.amazon.com/

Apache VCL (2016), http://vcl.apache.org, and http://vcl.ncsu.edu,

Appavoo, J. Uhlig, V., Stoess, J., Waterland, A., Rosenburg, B., Wisniewski, R., DaSilva, D., Van Hensbergen, E., Steinberg, U, 2010. "*Providing a Cloud Network Infrastructure on a Supercomputer*", Science Cloud 2010: 1st Workshop on Scientific Cloud Computing, Chicago, Illinois.

Appavoo, J., Uhlig, V., Waterland, A., Rosenburg, B., Stoess, J., Steinberg, U., DaSilva, D., Wisniewski, B., IBM Research, 2008, http://researcher.watson. ibm.com/researcher/view_group.php?id=1326

Appavoo, J. Uhlig, V., Waterland, A., Rosenburg, B., DaSilva, D., Moreira, J., 2009. "Kittyhawk: Enabling Cooperation and Competition in a Global Shared Computational System", *IBM Journal of Research and Development.*

Appavoo, http://kittyhawk.bu.edu/kittyhawk/Demos.html

Cloud Foundry Warden documentation http://docs. cloudfoundry.org/concepts/architecture/warden.html 2012.

Comet Cloud, (http://cometcloud.org)

Dreher, P., Mathew, G., 2014. Toward Implementation of a Software Defined Cloud on a Supercomputer. *IEEE Transactions on Cloud Computing IEEE 2014 International Conference on Cloud Engineering*, Boston, Massachusetts.

Dreher, P., Scullin, W., Vouk, M., 2015. "Toward a Proof of Concept Implementation of a Cloud Infrastructure on the Blue Gene/Q", *International Journal of Grid and High Performance Computing*, **7**(1), 32-41.

Felter, W., Ferreira, A., Rajamony, R., Rubio, J., 2014 *"An updated performance comparison of virtual machines and Linux containers,"* IBM Research Report, RC25482.

Graber, S., et. al,. LXC—Linux Containers. https://linuxcontainers.org/.

Higgins, J., Holmes, V., Colin, V., 2015 "Orchestrating Docker Containers in the HPC Environment", High Performance Computing, Vol 9137, *Lecture Notes in Computer science*, pp.506-513.

Hykes, S., et. al., *What is Docker*? https://www.docker.com/whatisdocker/.

Jacobsen, D., Canon, S., 2015. "*Contain This, Unleashing Docker for HPC*", Cray User Group.

Kim, H., Parashar, M., "CometCloud: An Autonomic Cloud Engine." *Cloud Computing: Principles and Paradigms.,* R. Buyya, J. Broberg, and A. Goscinski, eds., Wiley, 2011, pp. 275-297.

Marmol, V., et. al. Let me contain that for you: README. https://github.com/google/lmctfy/blob/ master/ README.md

NERSC, https://www.nersc.gov/

M. Parashar et al., Cloud Paradigms and Practices for CDS&E, *Research Report, Cloud and Autonomic Computing Center*, Rurgers Univ., 2012.

Penguin Computing http://www.penguincomputing.com/

Poettering, L., Sievers, K., Leemhuis. T., 2012. Control centre: The systemd Linux init system. http://www.h-online.com/open/features/Control-Centre-The-systemd-Linux-init-system-1565543.html

Shae, Z, et. al., "On the Design of a Deep Computing Service Cloud", RC24991 (W1005-053), 2010

Solomon, et. al. What is Docker? https://www.docker.com/whatisdocker/ .

Soltesz, S., Potzl, H., Fiuczynski, M., Bavier, A., Peterson, L., 2007 Container-based operating system virtualization: A scalable, high-performance alternative to hypervisors. *In Proceedings of the 2nd ACM SIGOPS/EuroSys European Conference on Computer Systems 2007*, EuroSys '07, pages 275–287.

Vouk, M., 2008, "Cloud Computing – Issues, Research and implementation", *Journal of Computing and Information Technology,* Vol 16(4), pp. 235-246.

Vouk, M., Rindos, A., Averitt, S., Bass, J., Bugaev, M., Peeler, A., Schaffer, H., Sills, E., Stein, S., Thompson, J., Valenzisi, M., 2009. "Using VCL Technology to Implement Distributed Reconfigurable Data Centers and Computational Services for Educational Institutions," *IBM Journal of Research and Development,* Vol. 53, No. 4, pp. 2:1-18.

Vouk, M., Sills, E., Dreher, P., 2010, "Integration of High Performance Computing Into Cloud Computing Services", *Handbook of Cloud Computing*, ed. Furht and Escalante.

K. Yelick et al., *The Magellan Report on Cloud Computing for Science*, US Dept. of Energy, 2011; www.nersc.gov/assets/StaffPublications/2012/MagellanFinalReport.pdf.