Reconfigurable Cloud: Consolidating HPC and Commodity Applications in the Cloud

Sumit R. Joshi, Vineeta Tiwari, Manish Kumar Abhishek

***Abstract*— Cloud computing is emerging as an additional deployment option for some of the HPC applications. But still, it currently does not come without drawbacks for application performance. These performance issues take place due to resource contention, where one VM is able to impact the performance of another. This interference directly creates impact on performance and scalability of HPC application. Additionally, virtualization overhead is also a barrier for adoption of cloud for HPC application.**

In this paper we present our work, Reconfigurable Cloud, cloud which is able to accommodate commodity application and HPC application on the same hardware. It solves the above issues by isolating HPC workload from other users and by giving near native performance for running HPC applications. With reconfigurable approach, same web-oriented cloud infrastructure can also be reused for HPC as well.

***Index Terms*—cloud, container, cgroup, HPC, Performance isolation, virtual cluster**

I. INTRODUCTION

High Performance Computing (HPC) allows engineers to solve data analytics, huge computations tasks and some complex problems that require high bandwidth, low latency networking, very high compute capabilities. Typically, to access shared HPC cluster, scientists and engineers must wait long in the queues. Else, they can acquire expensive hardware system, which leads to high capital expenditure.

Cloud computing [1] is a model for convenient, on-demand access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, services, and software) that can be easily provisioned as and when required. Cloud computing has efficient resource utilization by leveraging Resource aggregation. Thus, allows users to scale up to solve bigger problems. It also enables the system software to be configured as needed for individual application requirements. Cloud computing can help research groups, by providing convenient access to HPC clusters without the need to purchase and maintain sophisticated hardware.

HPC is performance-oriented, whereas clouds are cost and resource-utilization oriented. Additionally, clouds have traditionally been designed to run business and web applications. Past studies have demonstrated that commodity interconnects and the overhead of virtualization and resource contention are major performance barriers to the adoption of cloud for HPC.

In this paper we introduce our work, Reconfigurable Cloud, cloud that is able to accommodate commodity cloud application and HPC application on the same hardware without any overhead.

We make the following contributions in this work :

- We adopted the new container based Virtualization technology in cloud and designed its core components (Container creation Script, Monitoring) for on demand resource provisioning (CPU,RAM).
- We propose an approach to run HPC and cloud on the same hardware with reduced virtualization overhead and improved performance isolation with cgroup.

The paper is organized as follows: Section 2 provides some related work Section 3 talks about the some background Section 4 details about reconfigurable cloud stack. Section 5 details the proposed model and architecture for the cloud. Section 6 concludes and tells about the future plan of the work.

II. RELATED WORK

Much research has been done on examining the feasibility of running HPC applications on commodity cloud architectures. Specially, [2,3] have all tried different things with running critical applications and benchmarks on the Amazon EC2 cloud. Largely, the results of these investigations have demonstrated that EC2 is unable to provide a completely satisfactory HPC environment. In particular for communication intensive applications due to its lack of a high speed interconnect.

In [4], they have evaluated on inclusion of HPC interconnect in commodity cloud architectures. They suggested that the vast majority of commodity workloads neither require nor would significantly benefit from such an interconnect. However, this is a not disabling issue for the prospects of HPC in the cloud; if the other technical challenges currently hampering the deployment of HPC applications in the cloud can be overcome.

Gupta et al. [5] evaluated application characteristics for their suitability in cloud domain. They presented that different applications display different characteristics that make them more or less suitable to run in a cloud domain. Applications

Manuscript received May, 2015.

***Sumit R. Joshi**, ME Scholar, Department of Computer Engineering, Gujarat Technological University, Ahmedabad, India.*

***Vineeta Tiwari**, Senior Technical Officer, C-DAC, Pune.India.*

***Manish Kumar Abhishek**, IT Manager (Infrastructure), RailTel Corp India Pvt. Ltd, Gurgaon, India.*

ISSN: 2278 – 7798

with non-intensive communication patterns are good candidates for cloud deployments. For communication-intensive applications, supercomputers remain the optimal platform, largely due to the overhead of network virtualization in the cloud.

Xavier MG et al. [6] have proved container-based virtualization as an lightweight alternative to hypervisors in HPC context. They have compared container-based virtualization (LXC) with Xen Hypervisor and container achieved near-native performance of CPU, memory, disk and network. HPC will only be able to take advantage of virtualization systems if the fundamental performance overhead (such as CPU, memory, disk and network) is reduced. In this paper, container based virtualization technology is adopted for building infrastructure of Cloud.

Other approaches similar to our work have appeared in Kocoloski et al. [7]. They have introduced dual stack approach, which partitions a single node such that both commodity and HPC applications can execute concurrently without negatively impacting the other's performance. They have used two Virtual Machine Manager (VMM)on single node for cloud and HPC. Different VMM prevents cross VM interference and isolates HPC workloads from other users.

III. BACKGROUND

Virtualization is the key technology of cloud computing which is used for on-demand resource provisioning and currently the most popular one is the hypervisor-based virtualization. Virtual Machines (VMs) are created and configured by hypervisors to deploy applications and handle computing tasks for cloud users. In VM-based infrastructure, to create a VM instance, a full operating system should be running to get the resource isolation. The operating system often consumes some extra resources and spends a long time to boot. To reduce the startup time, cloud providers should always keep a large number of spare virtual machines around for provisioning. Thus, resource management using VMs as the units is heavyweight and less feasible for the just-in-time scalability.

The container-based virtualization is an alternative virtualization solution for application deployment and computing tasks, which shares the same kernel of host to avoid hardware emulation, as shown in Fig. 1. Unlike hypervisor-based virtualization, containers are built upon the operating system images and share a safely set of system libraries, instead of running a virtual machine instance with a complete operating system. The main benefit of containers is that they are light weight and much more efficient in terms of physical resource utilization than virtual machine based hardware virtualization. In this paper, the containers are created by the open source Linux Container LXC [8].

Resource management in Container based Virtualization is normally done by Control Groups (cgroup) [9], allows you to allocate resources—such as CPU time, system memory, network bandwidth, or combinations of these resources—among user-defined groups of tasks (processes) running on a system. You can monitor the cgroups you configure, deny cgroups access to certain resources.

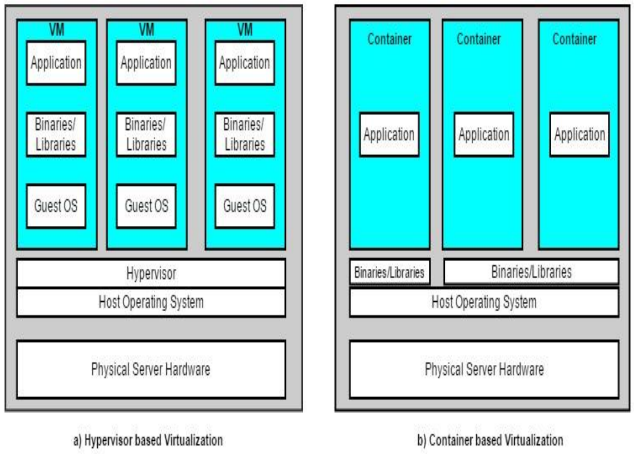

Figure 1 The Architecture of Virtualization Technologies

Cgroups are organized hierarchically, like processes, and child cgroups inherit some of the attributes of their parents. Multiple separate hierarchies of cgroups are necessary because each hierarchy is attached to one or more subsystems. A subsystem represents a single resource, such as CPU time or memory. Available Subsystems are:

A. *cpuset*

Assign physical CPU cores and memory node (e.g. on NUMA architecture) to a group. Useful parameter is:
cpuset.cpus- Set of CPU cores that can be accessed by a group of tasks.

B. *cpuacct*

Create a CPU resource usage report for each cgroups automatically. Useful parameters are:

- cpuacct.usage- CPU runtime used by all tasks in a group.
- cpuacct.stat- Divided cpuacct.usage between user and system.
- cpuacct.usage_percpu- Divided cpuacct.usage per CPU.

C. *memory*

Report memory usage and set physical memory limit for groups.Useful parameters are:

- memory.limit_in_bytes- Set the maximum value of physical memory for a group
- memory.stat- Report of memory statistics

IV. CLOUD ARCHITECTURE

With our cloud, user can create virtualized HPC cluster consisting of containers which are deployed on single resource pool. Unlike other approaches, we are using single resource pool for serving Cloud and HPC Request. User can specify the size of cluster, and is created on demand. Once job is finished, the cluster is destroyed and resources will be merged into resource pool.

Fig.2 describes the layered architecture of our proposed reconfigurable cloud. Physical Resources (compute, storage and network) are at the lowest layer of the stack, which are connected through 10Gb Ethernet. Above the hardware layer the host operating system is defined. Since our cloud will accommodate HPC application, performance of such application on such infrastructure will be of prime

ISSN: 2278 – 7798

importance. Thus, Container based virtualization is selected for near native performance.

A container will run on Host OS. Resource management will be handled by underlying OS and cgroup. The Cloud middleware stack is the core component that will be responsible for resource provisioning and scheduling, volume management, system monitoring for all the higher-level components and services.

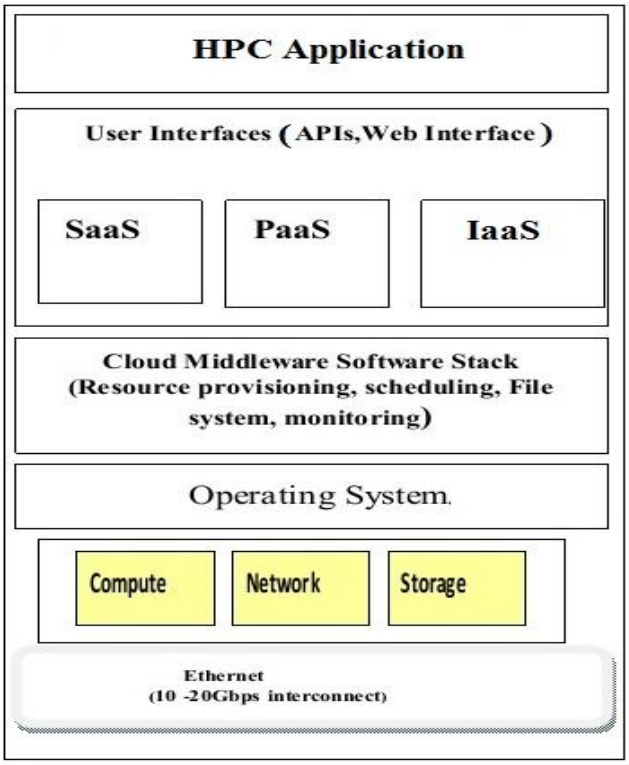

Figure 2 Reconfigurable Cloud Stack

V. Cloud Components

In our proposed solution we have two networks:

- Service network: In this network, actual compute node resides. Each node is configured with LXC, cgroup and libvirt [10]. VLAN is used to isolate different virtual clusters, running on these compute nodes.
- Storage Network: This network act as a node storage and image Repository for HPC and Cloud. In HPC case, separate master node HPC image and worker node HPC image is maintained.

Fig. 3 shows the components of reconfigurable cloud. The cluster and job management module acts as a controller to various sub components and builds a virtual cluster. It deals with Container Creation Script, a Monitoring module and batch system for scheduling jobs.

On receiving user request (i.e. memory and CPU core requirement), cluster and job management module will create the virtual cluster by leveraging container creation script. Submitted jobs are stored on NFS storage, where they are shared among the containers within the same virtual cluster.

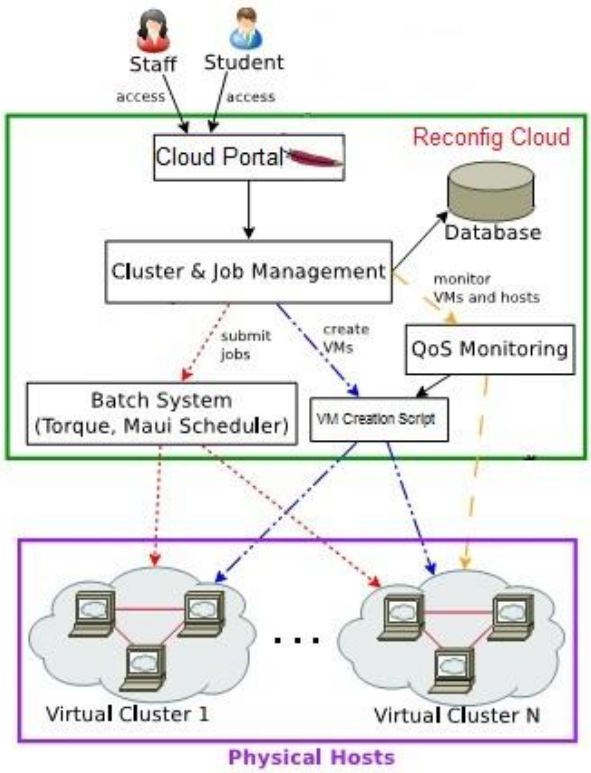

Figure 3 Architecture of Reconfigurable Cloud

A. Container Creation Script

Container creation script is responsible for finding available resources, creating containers (HPC or cloud) based on a master image, and deploying image to the physical host. As user specifies the cluster size, resources will be allocated based on availability. Each container has a unique name, a unique IP address, a unique MAC address, and a virtual network ID. As shown in Fig. 4, each virtual network has fixed set of generated IP-MAC pair addresses. Each virtual cluster has its own VLAN ID, in order to isolate from other virtual clusters.

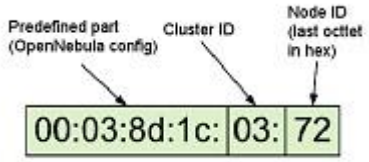

Figure 4 A generated MAC address of a virtual cluster

B. Monitoring

Quality of Service (QoS) will give performance assurance for individual application. It collects information about CPU load and memory usage. By monitoring these QoS parameters, cloud provides flexibility for adding more resources or containers during a peak load. It acts as a single point for monitoring and managing the cloud resources. The following features are supported:

- Monitors single node for identifying resource usage of commodity and hpc applications running on that node.
- Resource inventory lookup

ISSN: 2278 – 7798

C. Cloud Portal

Cloud Portal will be used for requesting and accessing the on demand virtual cluster or commodity cloud container. Fig. 5 shows the CLI client for our cloud.

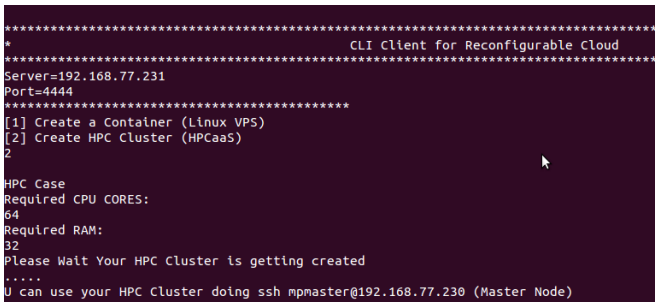

Figure 5 CLI for Reconfigurable Cloud

D. Performance Isolation with cgroup

When HPC application and commodity applications runs on the same system resource contention happens. It severly impacts HPC application performance and scalability. So performance isolation is required ,which is achieved by using cgroup. We are partitioning the system resources and assigning them to HPC and commodity applications. In our case, cgroup is used in following manner:

- Reserving Physical Memory space: As shown in Fig. 6, certain memory portion is reserved for HPC and certain one is for Cloud. Memory subsystem (memory.limit_in_bytes) is implemented.

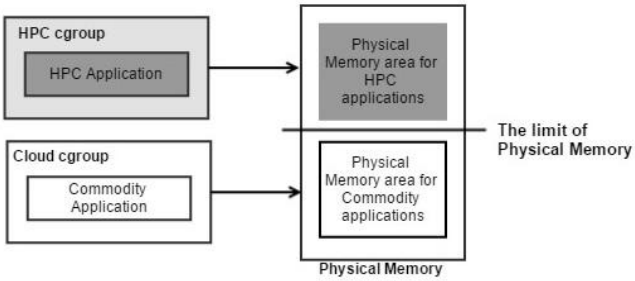

Figure 6 Reserving physical memory space for HPC and Cloud

- Monitoring Groups: We are monitoring HPC and Cloud cgroups by using special cgroup subsystem e.g. freezer cpuacct memory perf_event.

- CPU affinity / Exclusive possession of physical CPU core: HPC application use several physical CPU exclusively using cpuset.cpus and cpuset.cpu_exclusive to achieve short response time. As shown in Fig.7, CPU 0 and 1 is used by HPC application and CPU 2and CPU3 is used by commodity application.

So, HPC partition is able to isolate its resources (CPU,RAM) from competing loads on the rest of the system thereby preventing commodity workloads from interfering with HPC applications.

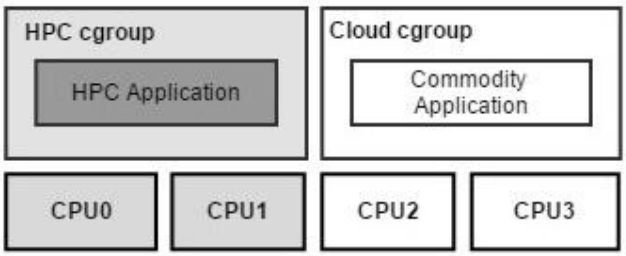

Figure 7 CPU affinity for HPC and Cloud

VI. Conclusion

HPC will adopt Cloud only if there is no overhead and no resource contention. Our work is an attempt to solve the above two issues. We have used Linux Container for achieving near native performance. While, Performance isolation is achieved by leveraging cgroup.

Our container based cloud, consolidates HPC application and commodity application and runs them on the same hardware. With our approach, the same hardware of a web oriented cloud infrastructure can be reused for HPC. It also leads to optimal utilization of IT infrastructure, by running both type of workload on the same infrastructure.

REFERENCES

[1] "Cloud Computing",2010. [Online]. Available: http://www.nist.gov/itl/cloud/

[2] Gupta, Abhishek, and Dejan Milojicic. "Evaluation of hpc applications on cloud." *Open Cirrus Summit (OCS), 2011 Sixth*. IEEE, 2011.

[3] Andrew J. Younge, Robert Henschel, James T. Brown, Gregor von Laszewski, Judy Qiu, Geoffrey C. Fox. "Analysis of Virtualization Technologies for High Performance Computing Environments", *IEEE 4th International Conference on Cloud Computing*, July 2011

[4] Hillenbrand, Marius, et al. "Virtual InfiniBand clusters for HPC clouds."*Proceedings of the 2nd International Workshop on Cloud Computing Platforms*. ACM, 2012.

[5] "The Who, What, Why and How of High Performance Computing Applications in the Cloud".[Online]. Available: www.hpl.hp.com/techreports/2013/HPL-2013-49.pdf

[6] Xavier, Miguel G., et al. "Performance evaluation of container-based virtualization for high performance computing environments." *Parallel, Distributed and Network-Based Processing (PDP), 2013 21st Euromicro International Conference on*. IEEE, 2013.

[7] Kocoloski, Brian, Jiannan Ouyang, and John Lange. "A case for dual stack virtualization: consolidating HPC and commodity applications in the cloud." *Proceedings of the Third ACM Symposium on Cloud Computing*. ACM, 2012.

[8] "Linux Container",2013. [Online]. Available: ww.linuxconatiners.org

[9] "cgroup",2009. [Online]. Available: https://www.kernel.org/doc/Documentation/cgroups/cgroups.txt

[10] "libvirt",2007. [Online]. Available: www.libvirt.org

[11] Doelitzscher, Frank, et al. "Viteraas: Virtual cluster as a service." *Cloud Computing Technology and Science (CloudCom), 2011 IEEE Third International Conference on*. IEEE, 2011.

Sumit R. Joshi , Pursuing the Masters degree in the field of Computer Engineering (IT Systems and Network Security) from the Gujarat Technological University, Ahmedabad.

Vineeta Tiwari, Senior Technical Officer, C-DAC, Pune.

Manish Kumar Abhishek, IT Manager (Infrastructure), RailTel Corp India Pvt. Ltd, Gurgaon.

ISSN: 2278 – 7798