International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 3 Issue 11, November-2014

Study of Virtualization Technologies in High Performance Computing

Mr. Amin Nazir Nagiwale
Computer Engineering
L.T..C.E. Navi Mumbai,India

Mr. Manish R. Umale
Computer Engineering
Lokamanya Tilak College of Engineering,
Navi Mumbai,India

Mr. Aditya Kumar Sinha
CDAC Pune, India

***Abstract*—Virtualization techniques gaining lots of attention in recent year and are mainly used to provide server consolidation ,hardware independence, resource isolation. Through requirements of HPC environment are different virtualization in HPC benefits in easy sharing of resources between both data intensive and CPU intensive jobs ,Dynamic allocation of resources , also helps to solve one of the major problem of HPC by providing easy checkpointing of jobs ,frees user from complex responsibilities. Therefore Virtualization actsas building block of cloud computing environment .**

Visualization solutions are available in both software (Hypervisor-based and Container-based systems) and hardware form(such as Intel-VT[5] and AMD-V [6]).Hypervisors (Xen [7], VMware and KVM [3]) are popular and fit best for many usage scenarios, but there are certain scenarios like resource sharing [1] and Custom environments [1] that require system virtualization with high degrees of both isolation and efficiency. Examples include HPC clusters, We present an alternative to hypervisors that is better suited to such scenarios i.e. container-based virtualization implementations (such as Linux-VServer[3] , OpenVZ [8] and Linux Containers (LXC) [4]) offer a lightweight virtualization layer, which promises a near-native performance.

Design principles of MapReduce framework like Moving computation , write once read many (WORM) access model use of commodity hardware with the assumption hardware will fail on regular basis leads enterprise computing environment more looks like HPC enviroment.YARN[11] and Mesos[12] which uses container-based virtulization (like Linux-container(LXC)) and growth in hardware (like multicore technology) adds more HPC capacities to the Hadoop enviroment so Hadoop can be considered as subset of HPC

Keywords—Big Data,HPC,YARN,Mesos

I. INTRODUCTION

Internet of Things (loT) [9] which connects objects like smart phones,computers,sensors all over the globe through Internet. As devices are connected to through Internet provides large number of services and produce huge amount of data Such environment resembles requirement of Cloud computingenvironment which provides access to shared resources (like servers,network,storage,application) on-demand basis. Cloud computing platform helps us to connects things around the globe so that we can access them at anywhere ,any time.

Applications that are running on cloud have special requirements like massive storage (i.e. distributed file system),real time data processing and analysis. As individual companies often generates petabytes or more of data and information is asset to such companies for continued growth and success. However, there are some problems like storage ,processing and analyzing of huge quantity of data, also most of the data available is in unstructured format which not useful for systems which requires structured format and also the hardware needed for traditional analysis is just too costly.

High Performance Computing (HPC) Clusters with visualization might to better solution to fulfill such requirements. but use of visualization is avoided in High Performance Computing (HPC) due to traditional overhead of visualization techniques such as hypervisor-based systems [VMware].but Container-based systems like Linux VServer,OpenVZ,,LXC considered as lightweight alternative with respect to resource sharing and performance isolation.

As Hadoop evolved as a distributed software platform for managing and transforming large quantities of data, and has grown to be one of the most popular tools. Therefore Hadoop with its core components like HDFS,MapReduce and High Performance Computing (HPC) clusters can be used to meet many of the needs of cloud computing environment in a cost-effective manner. Therefore Hadoop[10] on cloud computing has allowed on-demand computing capacity.

Experiments conducted in [1] effectively compares the container-based systems like Linux Containers (LXC) [4] ,Linux VSever and OpenVZ [8] when running on MR Clusters and experiment suggest that LXC performance well regarding to the performance isolation which is used by most recent systems like YARN[11] and Mesos[12].

In this work we would like to study YARN(Yet Another Resource Negotiator)[11] and Mesos [12] frameworks whichcome up with build in feature of container-based

virtualization to share the cluster among different applications .

In this paper Section II provides an brief overview of Hadoop which is an open source MapReduce platform to store and analyze massive amount of both structured and unstructured data .Section III overview of current container-based system Linux Containers(LXC)[4]. Section IV overview of YARN[11] also known as next generation Hadoop.Section V gives overview of Mesos [12] which is a cluster manager. Section VI describes comparison between Hadoop and HPC clusters.Conclusion and future work are presented in section VII.

II. MAPREDUCE

Map/Reduce is a open source distributed computational framework, originally designed to run on a large cluster of commodity servers and to scale to hundreds or thousands of nodes.

In MapReduce Environment same node can work as both compute and storage nodesi.e. based on the locality of data computational tasks scheduled and runs on the same set of nodes that hold the data required for the computations. MapReduce framework responsible for managing the hardware failure, job/task malfunction also manages the parallelismby splitting the input data into chuncks.converts these limited sized chucks also called as "splits" into intermediate key-value pairs which will be input for a set of Map tasks then it combines each keys value from output of Map step and processes the key/values as output data for set of Reduce tasks.

III. LXC (LINUX CONTAINERS)

Container-based Virtualization implementation such as Linux Containers (LXC)[4] is a operating system level virtualization architecture which is lightweight alternative to hypervisor-based architecture. As it provides virtualization at operating system level all containers share the same operating system kernel and provides weaker isolation than hypervisor-based virtualization.

Resource isolation is provided by implementing kernel namespaces like Process IDs (PID) ,File system , Inter-process communication (IPC) and Network namespace.

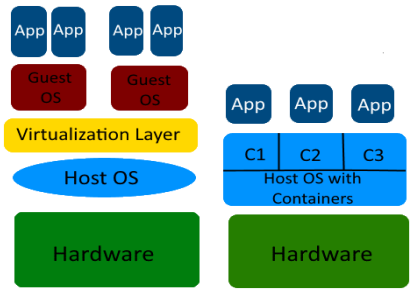


Fig .Hypervisor and Container based virtualization

a . Block IO Controller

cgroups which is the partitioning of the processes and all its children into hierarchical groups. These hierarchical groups will help in tuning the available system resources(i.e. Hardware and network)

Control Group Subsystems are classified in two parts

I. Isolation containers :cpuset, freezer, devices, checkpoint/restart

II. Resource controllers :cpu (scheduler), cpuacct, memory, disk I/O, network

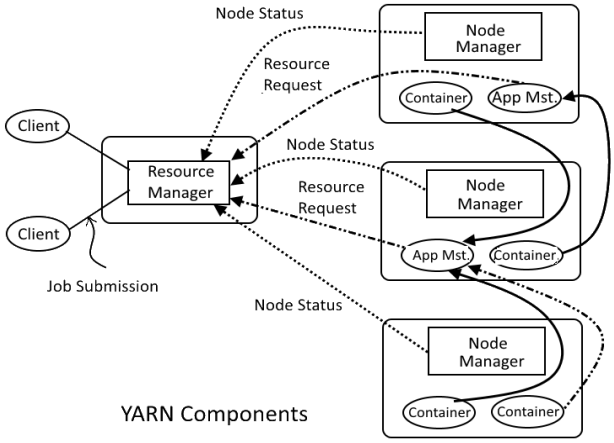
IO control policies are applied to both leaf nodes at intermediate nodesand same cgroup based management interface is used for blkio controller user options are used to switch the IO policies.

Two IO control policies are currently implemented first is proportionalweight time based division of disk policy as it is implemented in CFQ it is effective only on leaf nodes and second oneis throttling policy which specifies upper IO rate limits .

IV.YARN

YARN overcomes the short comings of Hadoop 1.0 by decoupling the programming model from the resource management and centralized control of MapReduce job's life cycle. This separation provides data processing model that is more than just a MapReduce . Spark[13],Storm[14],Giraph [15] are the examples of Programming models that may runs on the top of YARN.

To support Programming Model Diversity, Locality awareness, High Cluster Utilization YARN splits two important functionalities of JobTraker,resourse management and job monitoring into separate daemon. The basic idea is to have a per-cluster ResourceManager (RM) which handles recourse utilization,node's life cycle management and resolve conflicts between the tenants and per-application a ApplicationMaster (AM) which coordinates the logical plan of a job by sending resource requests to RM, and generates a physical plan from the resources allotted by RM.



a. ResourseManager

The RM runs as a daemon on one of the dedicated machine in a cluster, and arbitrates resources among various competing applications in the cluster i.e. acts as the central authority . this gives global view of the cluster resources, by which YARN maintains fairness and locality awareness across tenants.

Depending on the application requirements, priorities of scheduling ,and resource availability, the RM dynamically allocates containers to applications to run on particular nodes.containers (like LXC[2],KVM[2]) provides an operating system-level virtualization for running multiple isolated Linux systems (containers) on a single host. In this context container is logical unit which has its own its own process, network and storage space. To keep the track of containers assignment to the nodes and to maintain the global view of the cluster RM communicates with NodeManager(NM) which the daemon running on each node. Communication between RM and NM is based on Heartbeat[21].NM is responsible for monitoring recourses available on node and container life cycle management.

b. ApplicationMasters

The AM periodically heartbeats to the RM to provide information about its status and as applications resource request may shrike or grow dynamically it also request to update the record of its demand.

ApplicationMasters prepares one or more ResourceRequests which specifies there need

I. number of containers required (e.g. 500 containers)

II. resources (e.g.Memory,Number of CPU) per container

II. locality awareness/preferences

IV.priority of requests within the application running in container

In response to the heartbeats, the RM will allot a container lease on resources (like cpu,disk.network etc) available on particular node in the cluster. Based on the container it received from RM,AM will change the application's execution plan to balance the applications demands and available resources.

For MapReduce applications AM optimizes among map tasks based on locality. Based on the replication present in HDFS,AM assigns map task with input data closer to the container.Success or failure of container is determined by AM that is based on exit status reported by the NMs through RM.

c. Node Manager (NM)

After authenticatingcontainer leases ,NM manages the container life cycle (i.e. starting, killing the container).NM registers itself with RM,and heartbeats[21] its status.

In YARN all containers including ApplicationMasters(AM) is represented by Container Launch Context (CLC)[19]. which includes environment variables required to configured, dependencies information about remotely accessible storage, the command necessary to create the process when the container is starts its execution.NM first authenticates container lease and then configures the environment for the container. Before launching the container NM copies all dependencies like packages, executables to the local node's storage .Also periodically garbage collects the dependencies that are not required by running containers.

V. MESOS

Mesos [20] is a plateform that uses cotainers to isolate resourses among the frameworks like hadoop,MPI. Mesos [20] is a platform which shares clusters of commodity machines between multiple different cluster computing frameworks, such as Hadoop and MPI.Mesos provides performance isolation between different framework executors running on the same slave by using existing OS isolation mechanisms.Mesos adds and removes resources from an executor as it starts and finishes tasks.

VI. DIFFERENCE BETWEEN CONTAINER-BASED AND HIGH PERFORMANCE COMPUTING (HPC) CLUSTER

we will distinguish both the basic of Hardware, Resource Scheduling, Parallel Computing Model

A. Hardware

Hadoop system is based on fundamental assumption that moving computation/code is cheaper than moving data i.e. it is more efficient to execute computations on nodes that locally store the data involved which allowsbetter performance on commodity clusters with relatively slow network fabrics (e.g. 1 GigE).

On the other hand HPC cluster ,distributed file system (PVFS[16],Lustre[17],GPFS[18]) and faster networking infrastructure

typically available in an HPC environment which enables maximum overall Hadoop performance than moving the computation.

HDFS file system does not support POSIX standards that is it is non POSIX -compliant file system, and once data is writtenit is non-modifiable.HDFS supports**write-once, read-many access (WORM)**model. HDFS replicates data blocks across multiple devices. Default replication facter is three, "local", “nearby”, and “far away”

Whereas HPC file systems (PVFS[16],Lustre[17],GPFS[18]) are POSIX compliance i.e. most operations are atomic and clients are not able to see the stale data or metadata

II. RESOURCE SCHEDULING

Resource management is one of the major difference between Hadoop and HPC systems.

Two core components of Hadoop, job tracker and Tasktracker are responsible for resource management. Jobtracker handles the all jobs submitted by the client and makes all scheduling decisions. It also monitors all running tasks and will restart or kill the tasks in event of hardware failure .It uses timeout strategy to detect the hardware failure reassign tasks. If incomplete Map or Reduce task exceeds the specified timeout, it assumed that node is fail and restart the task on same node or assigns the task to the other node having input data.

on the other hand HPC resource scheduler has fine-grained control over the resources like memory, cores, time that are assigned to the users application. Load Leveler, Grid Engine, Moab are the tools used to assign the resources. To recover from the hardware failure HPC system uses check

pointing if job fails due to a power failure, node crashing, job malfunction ,job resumes from last successful checkpoint. It must restart the job if it has not been checkpointed

III. PARALLEL COMPUTING MODEL

As Hadoop is designed with fundamental assumption of "moving computation" therefore MapReduce algorithms are considered as generalization of Single instruction, multiple data (SIMD) operation where single instruction or all instruction of single problem are large dataset.

MapReduce programming model is inspired functional programming. As with functional languagesmap and reduce functions cannot change the input data, this restriction provides high level of scalability and redundancy.

Parallelism isachieved by breaking data into independent parts with no side effects during map step that map step do not modify any data and reducer applies same reduction process to result set of Map process

Whereas in HPC cluster, jobs are SIMD and MIMD (multiple-instruction, multiple-data) and

programmer is responsible for determining how to execute the parallel algorithm

this provides flexibility which comes with addition responsibilities. On the other handin Hadoop users only defines Map and Reduce functions without any restriction

VII. CONCLUSION

CPU and I/O overhead of server/machine is eliminated by using virtualization techniques like Para-virtualization ,full virtualization, operating system virtualization. In context of High Performance Computing (HPC) only operating system virtualization support the requirements like isolated multiple user-space instances, easy customization of environment, faster network infrastructure of HPC computing with minimal overhead.

With advancement in Hadoop ecosystem and commodity hardware, it can be considered as subset of HPC. as With MapReduce framework user only need to define Map and Reduce steps complex tasks like defining parallelism, resource management, job or node recovery are encapsulated into MapReduce Framework.

Linux Containers (LXC) when running on MR Clusters provides better resource isolation and YARN and Mesos described in sections III and IV are benifited from container-based systems provides more HPC capacities to the Hadoop cluster.

REFERENCES

[1] Miguel G. Xavier, Marcelo V. Neves, Cesar A. F. De Rose "A Performance Comparison of Container-based Virtualization Systems for MapReduce Clusters" 2014 22nd Euromicro International Conference on Parallel, Distributed, and Network-Based Processing

[2] Miguel G. Xavier, Marcelo V. Neves, Fabio D. Rossi, Tiago C. Ferreto, Timoteo Lange, Cesar A. F. De Rose "Performance Evaluation of Container-based Virtualization for High Performance Computing Environments" 2013 21st Euromicro International Conference on Parallel, Distributed, and Network-Based Processing

[3] N. Regola and J.-C. Ducom, "Recommendations for virtualization technologies in high performance computing," IEEE Second International Conference on, 2010

[4] LXC [Online] Available:http://en.wikipedia.org/wiki/LXC

[5] Intel-VT [Online] Available:http://www.intel.in/content/dam/www/public/us/en/documents/product-specifications/vt-directed-io-spec.pdf

[6] AMD [Online] Available:http://en.wikipedia.org/wiki/X86_virtualization

[7] Xen [Online] Available:http://en.wikipedia.org/wiki/Xen

[8] OpenVZ [Online] Available:http://openvz.org/Main_Page

[9] Prahlada Rao B. B, Payal Saluja, Neetu Sharma, Ankit Mittal, Shivay Veer Sharma "Cloud Computing for Internet of Things & Sensing Based Applications"

[10] Hadoop [Online]. Available: http://hadoop.apache.org

[11] YARN [Online]. Available: http://hadoop.apache.org/docs/current/hadoop-yarn/hadoop-yarn-site/YARN.html

[12] Apache Mesos [Online]. Available: http://mesos.apache.org/

[13] Apache Spark [Online] Available:https://spark.apache.org/

[14] Apache Strom [Online] Available:https://storm.apache.org/

[15] Apache Giraph [Online] Available:http://giraph.apache.org/

[16] Parallel Virtual File System [Online] Available:http://en.wikipedia.org/wiki/Parallel_Virtual_File_System

[17] Lustre (file system) [Online] Available:http://en.wikipedia.org/wiki/Lustre_(file_system)

[18] General Parallel File System [Online] Available:http://en.wikipedia.org/wiki/IBM_General_Parallel_File_System

[19] Node Manager [Online] Available:http://docs.hortonworks.com/HDPDocuments/HDP2/HDP-2.1.5/bk_using-apache-hadoop/content/node_manager.html

[20] Mesos: A Platform for Fine-Grained Resource Sharing in the Data Center

[21] Heartbeat (computing) [Online] Available:http://en.wikipedia.org/wiki/Heartbeat_(computing)