Semi-shared storage subsystem for OpenNebula

Sándor Ács, Péter Kacsuk, Miklós Kozlovszky
MTA SZTAKI Computer and Automation Research Institute,
H-1518 Budapest, P. O. Box 63, Hungary
[acs, kacsuk, m.kozlovszky]@sztaki.hu

***Abstract*— To address the limitations of OpenNebula storage subsystems, we have designed and developed an extension that is capable of achieving higher I/O throughput than the prior subsystems. The semi-shared storage subsystem uses central and distributed resources at the same time. Virtual machine instances with high availability requirements can run directly from central storage while other virtual machines can use local resources. As I/O performance measurements show, this technique can decrease I/O load on central storage by using local resources of host machines.**

Keywords - cloud computing; OpenNebula; storage subsystem; I/O performance.

I. INTRODUCTION

Cloud computing opens a new way of thinking about distributed information technology (IT) infrastructures [1]. The paradigm is based on virtualization technologies (server, storage, network, etc.) and it uses multiple experiences gathered from grid and cluster computing as well. In the three layered cloud model (Software/Platform/Infrastructure as a Service), the IaaS is the bottom layer that provides fundamental computing resources to consumers [2]. IaaS can be built from traditional IT hardware components and cloud middleware software.

OpenNebula [3] is an open source software stack, born from a research project and became one of the best-known IaaS cloud solution. The main components of OpenNebula are the front-end, compute nodes, image repository and networking infrastructure. The front-end machine is responsible for the core services (user authentication, scheduling, etc.) and provides an entry point for consumers. Compute nodes are hosts of virtual machines (VMs). The image repository handles virtual disk images and its storage subsystem contains physically the images. Compute nodes reach disk images directly via shared storage or copied through the network. If compute nodes use shared storage, VMs will consume the same resource that can cause decreased I/O performance for VMs. If compute nodes use non-shared storage, they will suffer from some disadvantages (e.g., slower VM deployment).

There are several open issues in cloud computing and one of them is related to the virtualized I/O performance [4]. Related studies [5] expose that the storage subsystem can play the key role from efficiency point of view in a cloud.

The main contribution presented in this paper is the concept of semi-shared storage subsystem that tries to alleviate the negative effects and find a trade-off between shared and non-shared storages. The semi-shared storage subsystem can provide benefits from both of the storage subsystems at the same time. It can share disk images between compute nodes for fast and flexible deployment and it can decrease the load with distributed non-shared resources.

We designed, implemented and tested the semi-shared storage subsystem for OpenNebula. I/O performance of the prototype is investigated in a local cloud installation and its values are compared to results of other existing storage subsystems. We present a technique that is able to achieve higher I/O throughput in OpenNebula than its prior solutions.

This paper is organized as follows: first, we introduce the related research results in Section II. Then, we present image management and features of the storage subsystems in OpenNebula in Section III. Next, we detail the semi-shared storage subsystem that helps to reduce the load in a cloud infrastructure. Afterwards, in Section V, we present the test infrastructure and results of the performance benchmarks. In Section VI, a production use case is introduced. Finally, we conclude our research in Section VII.

II. RELATED WORK

As related works have been already started to investigate the I/O performance of cloud infrastructures. Goshal et al. [5] introduced the Magellan project that explored some IaaS clouds from High Performance Computing (HPC) suitability point of view. The paper discloses that the performance of communication intensive applications is degraded by the virtualized I/O subsystem. Benchmarks were used on different types of clouds (e.g., Amazon EC2) and compared the results with local infrastructure measurements. Their results pointed out the major performance bottleneck which can be caused by virtualized environment.

Lihtium [6], a distributed storage system, was designed in order to avoid the limitation of centralized shared storage systems of cloud infrastructures. This solution is complex and specialized for virtualization workloads aimed at the large-scale cloud infrastructures and data-centers. The semi-shared storage solution for OpenNebula is lightweight and it can enhance the I/O throughput in small and middle-scale cloud infrastructures as well.

Ousterhoutet et al. [7] presented that the disk-oriented storage systems are problematic in a dynamic cloud environment. A new storage system was designed in order to achieve lower access latency and higher bandwidth. The solution is based on the main memories aggregation of the nodes. This new approach called RAMCloud, where all

 ISBN: 978-1-61208-216-5

information (disk images as well) is kept in DRAM. This solution promises 100-1000x faster throughput than disk-based systems and 100-1000x lower access latency. Using RAM based storages for improving the I/O performance of clouds has many benefits, however traditional disk based storages cost much less for the same capacity.

Sheepdog [8] is a distributed storage system that is integrated into QEMU/KVM [8]. It provides block level storage volumes redundantly based on distributed resources. Sheepdog supports volume management features such as snapshot and it can be scaled up without single point of failure to several hundred nodes. However, it cannot guarantee high bandwidth and low latency storage.

III. STORAGE SUBSYSTEMS AND DISK IMAGES

The image repository, accessible by the compute nodes, serves as a store for disk images in IaaS. The compute nodes can create copies from the disk images or they can use the images directly in order to create virtual machine instances.

1) Storage subsystem

In OpenNebula, the compute nodes can reach the disk images in different ways: (i) via shared storage or (ii) by copying it through the network from the image repository.

a) Shared storage

In Fig. 1, a compute node and an image repository with virtual disk images can be seen, where shared storage is available from the compute node, which can start the virtual machine instances.

With shared storage, the VMs can be started without copying it through the network and live migration is available for instances. The live migration is a procedure when a VM instance is moved from one host to the other without outage which can be sensed by end users.

The disadvantage of the shared storage is that all of the deployed virtual machines could use the same resource (storage subsystem of image repository) concurrently. The decreased I/O throughput causes performance loss for VM instances.

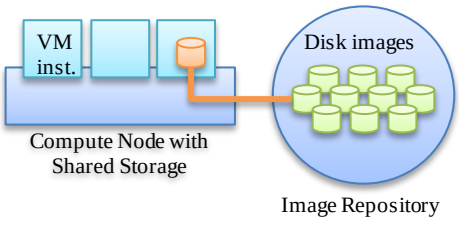

Figure 1. Compute node with shared storage

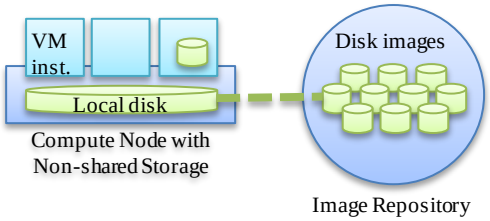

Figure 2. Compute node uses local copy from disk images

b) Non-shared storage

In Fig. 2, shared storage is not available, so the compute node cannot attach disk images directly from the image repository. The disk images should be copied through the network (broken line in Fig. 2) and stored in local storage. The virtual machine instances are created from local copies. The non-shared storage can cause peaks on I/O load while disk images are copying, however these peaks can be ignored if the VM instances are used long-term (days). In this paper, we investigate this option.

This storage subsystem can reduce the load on image repository with distributed resources however VM deployment and image sharing (copying and saving) takes more time and the live migration is not available.

2) Disk images

OpenNebula uses two types of disk images from volatility point of view. The state of the disk images can be persistent or non-persistent. If a virtual machine runs with persistent disk, the changes will be stored after shutdown. If a virtual machine uses non-persistent disk, the disk image will be deleted after shutdown.

Persistent and non-persistent disks can be used with shared- (Fig. 1) and non-shared (Fig. 2) storages as well. These options are detailed in the next two sections.

3) Disk images with shared storage

*a) Persistent disk:*The Virtual machines deployment is fast (compared to the overall process time of copy image from repository to local disk and from local disk until deployment). It is not needed to write back the changes after shut down because the disk image is attached directly to the image repository. The live migration is available in this option.

*b) Non-persistent disk:*These disk images are copied from the image repository, that takes more time than in option (a), however it is still faster than using non-shared storage. After shut down, the disk images are deleted, except if they were forced to be saved that means a copy from the instances in this case. The live migration is also available.

Summarized: With shared storage the fast VM deployment and live migration can be achieved. On the other hand, many VM instance with I/O intensive workloads can cause heavy load on the image repository.

 ISBN: 978-1-61208-216-5

4) Disk images with non-shared storage

*a) Persistent disk:*The Disk images are copied two times (for starting and saving) in the life of a virtual machine instance. The procedure of moving the VM instance from a compute node to another, takes more time than acceptable for live migration.

*b) Non-persistent disk:*These disk images are copied through the network from the image repository as well. They are deleted after shut down (except if they were forced to be saved by the user). The live migration is not available because of the non-shared storage.

Summarized: There is an overhead on the disk image sharing however the I/O workloads of the VM instances are distributed on the compute nodes. However, in this case just the slower cold migration is available instead of live migration for VM instances.

IV. THE SEMI-SHARED STORAGE SUBSYSTEM

As related works pointed out in Section II, the shared storage can be a bottleneck in a cloud and it can cause decreased I/O performance for VM instances. In this paper, we focused on the disk I/O. As presented in Section III, non-shared storage subsystem can be used to decrease the load on the image repository and to increase the VMs' disk performance because the VM instances use (distributed) local copies from the disk images instead of the shared storage.

In order to avoid the high load on image repository and increase the performance of the virtual disks, we propose the notion of semi-shared storage. As a proof of concept it was elaborated and implemented to OpenNebula.

The basic ideas were the following: the image repository component practically has more reliable storage subsystem than compute the nodes. Some VMs (e.g. database or firewall servers) may need to be migrated without outages. These VM instances should have persistent disk images based on shared storage because it takes time to copy the disk images trough the network and resume the operation of the VM instance. The loss of the fast start and live migration opportunities can cause that the non-shared storage is not sufficient to be used in high available production systems. However not all of the VM instances require features like live migration, fast deployment and having persistent disk images. These instances can be used with non-shared storage. (Of course, the non-persistent disk can be saved as well by the users.)

Our contribution to OpenNebula is the Semi-shared storage subsystem, which uses shared storage for persistent disk images and local copies with non-persistent disk images for creating VM instances. The benefit of this solution is that the shared- and non-shared files-systems can be used at the same time on the same compute node. The semi-shared storage subsystem can satisfy high availability requirements (like the original shared storage subsystem).

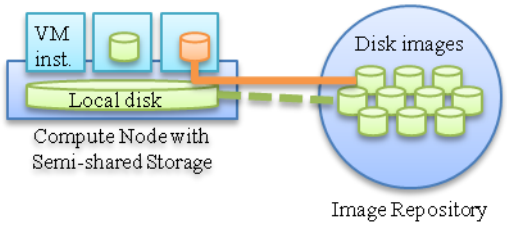

Figure 3. Semi-shared storage using local and shared resources concurrently

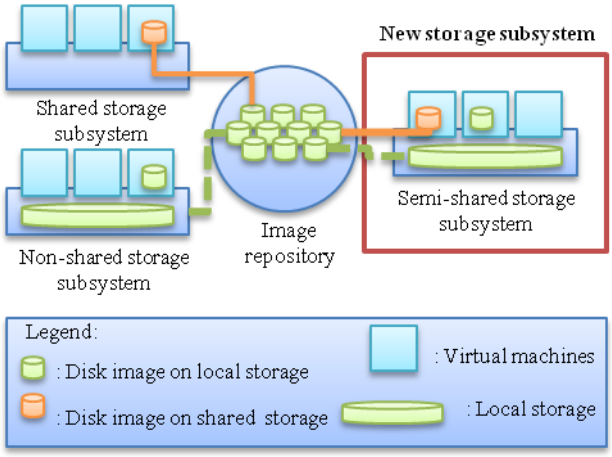

Figure 4. Semi-shared storage is using local- and shared resources at the same time

Moreover, it is able to decrease the load on image repository by using local storages of compute nodes. These may increase the performance of the disk images, especially in an over-provisioned cloud (like the original non-shared storage).In Fig. 3, the compute node uses shared- and non-shared storage at the same time. Shared storage is used for VM instances deployed with persistent disk image and local copies (non-shared storage) are used for VM instances deployed with non-persistent image. Fig. 4 summarizes the original (left side) and the new (right side) storage subsystems for OpenNebula.

V. TEST INFRASTRUCTURE AND PERFORMANCE BENCHMARKS

Experiments were carried out on an installation of OpenNebula (version 3.2) that consists of two compute nodes and one image repository. Technical details are summarized in "TABLE I".

TABLE I. CONFIGURATION OF THE TEST BED

Components of the test infrastructure				
Role	***Type***	***CPU***	***HDD***	***MEM***
Image-Repository Front-end	Sun Fire X2200 M2	2xQuad-Core Opteron 2.3G	Seagate ST32500N SATA	12G DDR2
2XCompute Nodes	Sun Fire X2200 M2	2xDual-Core Opteron 1.8G	WDC WD2500JS SATA 250G	8G DDR2

1) Testing of semi-shared storage

In order to prove that higher I/O performance is achievable by using the semi-shared storage subsystem than the prior (shared and non-shared) solutions could provide, I/O benchmarks were performed on the test cloud. The benchmarks were sequential read throughput tests because sequential read is a typical storage parameter [6]. At the same time, 8 exactly the same virtual machine instances were used to stress and load the I/O subsystem, while the performance was measured inside the virtual machine and directly on the physical block device with the iostat tool. Iostat is an I/O performance monitoring tool for Linux based systems. During the tests, the virtualization hypervisor was KVM and caches as well as buffers were disabled on every layer (files-system, hypervisor, etc.) for more accurate results [5]. The first diagram (Fig. 5) presents the results when a shared storage server and one compute node use semi-shared storage for benchmarking. The available I/O performance was measured in the image repository, compute node and individually in the VM instances as well. The benchmark values are the sequential read throughputs when all the 8 virtual machines are running. The first test batch has 9 pairs of columns. The pairs are distributions of VM instances between local and remote resources. In a pair, the first column is the aggregated I/O performance of the VMs and the second is the aggregated I/O performance of image repository and compute node.

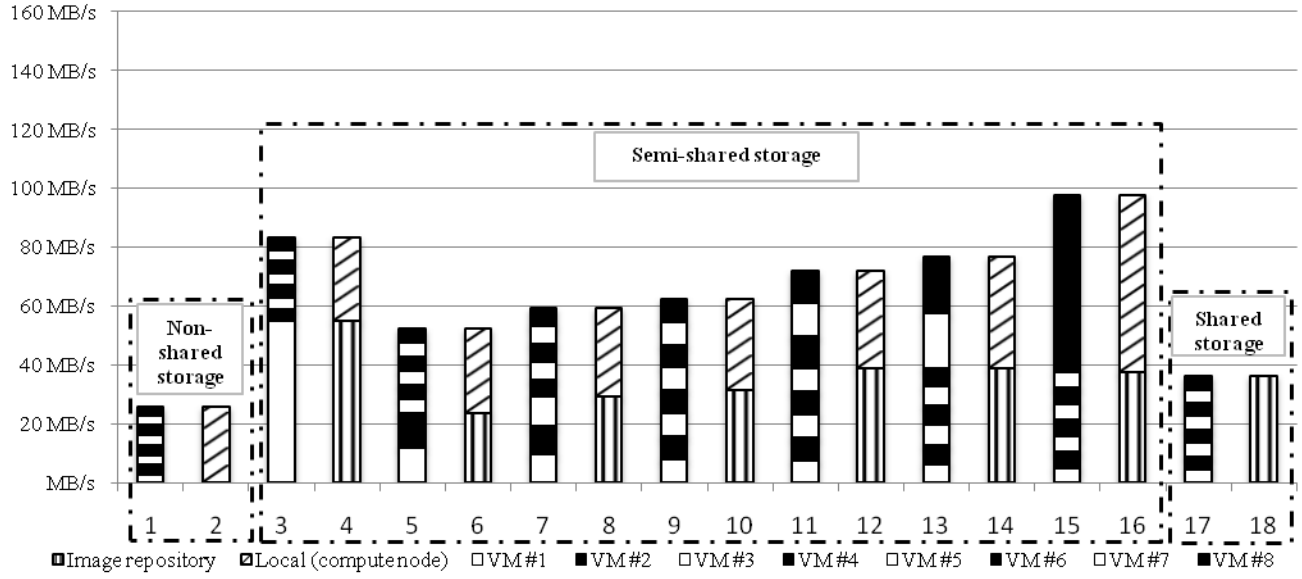

Figure 5. I/O benchmark with the image repository and one compute node

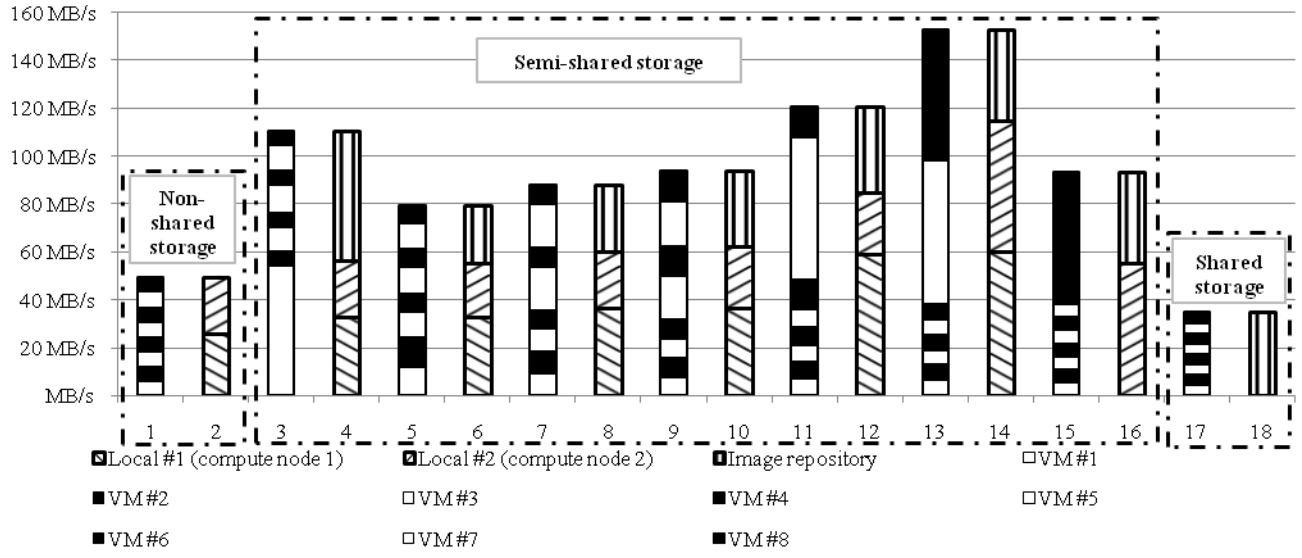

Figure 6. I/O benchmark with the image repository and two compute nodes

In the first test (column one and two), all of the VM instances are using the local storage of the compute node, which is special because it is the default distribution when non-shared storage subsystem is used. In the second test (7 VM instances running from local storage and one instance directly from image repository), the aggregated I/O performance is increased almost by three times compared to the first test, because the shared storage (image repository) was dedicated to only one VM. The last test in the batch is special as well, because all the VM instances running from image repository which is default when shared-storage subsystem is used for the OpenNebula cloud. The results show that the semi-shared storage subsystem can serve higher aggregated I/O performance than the original storage subsystem solutions in OpenNebula.

In the second test batch (Fig. 6), one image repository server (as shared storage) and two compute nodes are used and benchmarked. In the diagram, it can be seen that highest aggregated and individual (from VM instance point of view) I/O performance can be achieved if the VM instances can use exclusively a local- or the shared storage. If more computing nodes were added to test-bed, bigger performance gap would be measured between the shared- and semi-shared storage subsystems. These points to the fact that the non-persistent VM instances (running from local disk of compute nodes) are preferred and the number of the persistent VM instances (running from image repository) should be kept low if the image repository consists of a single machine. In this paper, we do not investigate and discuss the clustered or distributed storage technologies which can expand the capacities of the image repository.

VI. PRODUCTION USE CASE

Some early tests with OpenNebula showed us that I/O throughput can be problematic if VMs generate I/O intensive workloads. In order to protect our production services, we wanted to isolate production, developer and tester VMs. Separated clouds can be build for these purposes, however the utilization of the cloud components would be worse in that case. Some of our VMs require live migration which excludes to use non-shared storage subsystem for OpenNebula.

After tests were successfully running in the test-bed, semi-shared storage subsystem was put production in MTA SZTAKI. Our second cloud installation has 64 CPU cores, 152GB RAM and ~5 TB storage. Usually, there are 40-60 VM instances are running concurrently. ~10 instances of them are in production, about 20 instances are used by developers and the others are running for testing purposes. Production VMs are using only persistent disk images and the testing VMs are using only non-persistent disk images. (Developers are using both of them.) With this distribution, we managed to solve the high utilization of our resources without compromising the production services.

VII. FUTURE WORK

For IOPS-critical server workload, flash based storages are preferred to use, like SSDs or traditional DRAM [10]. We already have performed some experiments with VMs running in DRAM. We considered expanding OpenNebula with DRAM based storage solution and combined it with semi-shared storage subsystem as well. Going forward, we are planning to explore different file-system solutions for image repository. Ceph [11] could enhance scalability and provide software based redundancy for image repository of OpenNebula. In this paper, we did not investigate the I/O performance from a networking point of view, however we plan to explore the effects on the network by using different types of storage subsystem for OpenNebula.

VIII. CONCLUSION

In this paper, the performance of virtualized I/O subsystems was discussed, which is one of the most considerable limitations of cloud infrastructures. The investigation is focused on OpenNebula and its storage subsystem solutions. In this cloud middleware, we experienced scalability and I/O throughput problems. To relieve the problem, OpenNebula provides distributed storage option however fast VM deployment and live migration features are lost with that option. We presented the semi-shared storage subsystem that is able to achieve higher I/O throughput than other storage solutions do in OpenNebula by using central and local resources at the same time. Finally, test results and the production use case showed that we managed to expand I/O performance related bottlenecks in OpenNebula.

ACKNOWLEDGMENT

The authors would like to thank Márk Gergely, Péter Kotcauer, Zsolt Németh and Gábor Kecskeméti for their input and comments that helped to shape and to improve this paper. This work is supported in part by the EDGI EU FP7 project (RI-261556).

REFERENCES

[1] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic. Cloud computing and emerging itplatforms: Vision, hype, and reality for deliveringcomputing as the 5th utility. Future GenerationComputer Systems, 2009.

[2] P. Mell and T. Grance. NIST definition of cloud computing. National Institute of Standards and Technology. Oct. 2009.

[3] B. Sotomayor, R. S. Montero, I. M. Llorente, and I. Foster. Virtual Infrastructure Management in Private and Hybrid Clouds. IEEE Internet Computing, vol. 13, no. 5, pp. 14-22, Sep. 2009.

[4] V. O. Póserné: Comparing the webservers of the opensource and the closed source operation systems, in Proc of the 5th International Symposium on Applied Computational Intelligence and Informatics, Timisoara, Romania, 2009, pp. 169-172

[5] D. Ghoshal, R. S. Canon and L. Ramakrishnan. I/O Performance of Virtualized Cloud Environments. DataCloud-SC '11, Nov. 2011.

[6] J. G. Hansen and E. Jul. Lithium: virtual machine storage for the cloud. In SoCC '10: Proceedings of the 1st ACM symposium on Cloud computing, pp. 15–26, New York, NY, USA, 2010.

 ISBN: 978-1-61208-216-5

[7] J. Ousterhout , P. Agrawal , D. Erickson , C. Kozyrakis , J. Leverich , D. Mazières , S. Mitra , A. Narayanan , G. Parulkar , M. Rosenblum , S. M. Rumble , E. Stratmann and R. Stutsman. The case for RAMClouds: scalable high-performance storage entirely in DRAM, ACM SIGOPS Operating Systems Review, v. 43 n. 4, Jan. 2010.

[8] Sheepdog project. http://www.osrg.net/sheepdog/.

[9] A. Kivity, Y. Kamay, D. Laor, U. Lublin, and A. Liguori. KVM: the Linux virtual machine monitor. In Ottawa Linux Symposium, 2007.

[10] A. M. Caulfield , L. M. Grupp , S. Swanson. Gordon: using flash memory to build fast, power-efficient clusters for data-intensive applications. Proceeding of the 14th international conference on Architectural support for programming languages and operating systems, Washington, DC, USA, Mar. 2009.

[11] Sage Weil, Scott A. Brandt, Ethan L. Miller, Darrell D. E. Long, Carlos Maltzahn, Ceph: A Scalable, High-Performance Distributed File System, Proceedings of the 7th Conference on Operating Systems Design and Implementation (OSDI '06), Nov. 2006.

 ISBN: 978-1-61208-216-5