Comparison of Container Technology Systems for MapReduce Clusters of Different Cloud Providers

Enam Shah
Dept. of Computer Engg
Santa Clara University
Santa Clara, CA, USA
enamshah09@gmail.com

Parth Shah
Dept. of Computer Engg
Santa Clara University
Santa Clara, CA, USA
shahp13594@gmail.com

Balaji Sai
Dept. of Computer Engg
Santa Clara University
Santa Clara, CA, USA
balaji.sai16@gmail.com

Anuj Shah
Dept. of Computer Engg
Santa Clara University
Santa Clara, CA, USA
anujyshah@gmail.com

***Abstract*— Hadoop is a framework that supports the processing of large data sets in a distributed computing environment. To run Hadoop like application on Virtual Machine, there is always overhead of hypervisor which slows down the performance due to which they can't reach the native performance. Whereas containers being lightweight are a good alternative to Virtual Machine. In this work, we compared different cloud providers in the market like Amazon AWS, Google Cloud with native bare metal like Apple MAC in terms of execution and sensibility when running on MR clusters.**

Keywords— Container-based virtualization, MapReduce, Hadoop, MR Benchmarks.

I. INTRODUCTION

As data is growing day by day, the need of computing such large data is increasing simultaneously. MapReduce has turned into an extremely well known programming model for extensive data analysis. Data processing system such as Hadoop [3] uses MapReduce(MR) to simplify the large scale distributed systems and even offers better scalability and performance. Hadoop has beaten obstructions, taking care of overwhelming issues containing extensive volume of data. Hadoop is as now the most well-known and widely deployed of the open-source MR executions. It was fundamentally upheld by Yahoo! and later also used by many Internet companies, including Twitter, Facebook [4] and Amazon [5]

Similarly, now we can observe rapid growth of data in cloud computing. Clients are progressively utilizing cloud services to store a lot of data and exploit advantages for example adaptability, scalability, cost proficiency, reasonability, fault tolerance and greater computing and storage feasibility.

Computing is mainly supported by hypervisor-based virtualized system like Xen [6] or KVM [7]. Running Hadoop application on such systems creates an inherent performance overhead due to the extra layer of hypervisor in cloud computing. Therefore, running Hadoop applications on cloud based systems could not reach the native performance. Whereas container-based virtualization system provides a very good alternative to hypervisor-based virtualized system, providing near native performance and better manageability.

They are many providers we chose google and amazon, both these providers give container based service as Google container engine and Amazon AWS EC2 Container service. Whenever we launch a new container it is launched in VM instance but depending on the capacity of the VM, it can hold certain number of containers in it therefore, the containers which are launched on cloud providers have hypervisor overhead. As we can see, it defeats the purpose of container due to the extra layer of hypervisor. But since Docker container has many other features, it led us to use containers.

Some of the features are listed below:

1. Increased productivity: There are 2 main features the first is bringing the production environment as close as possible and second is making the development environment as fast as possible and interactive.
2. Rapid Deployment: Bringing up a new hardware in past would take few days, with virtual machine it came down to few minutes, but using containers it came down to few seconds. So, scaling your system is now faster than ever.

- We have compared the best two containers based virtualization systems i.e. Amazon AWS, Google Cloud.
- We have examined the Hadoop clusters over different situation on container-based virtualization system by utilizing distinct MR benchmarks.
- We have evaluated the cost and performance of each provider.

The approach to compare container is divided into single node and multi node cluster. In multi node cluster we have chosen Amazon EMR and Google DataProc service to compare.

A. Amazon EMR

Amazon EMR (Elastic MapReduce) gives a managed Hadoop system that makes it simple, quick, and cost-effective to handle large amount of data across dynamically scalable instances of Amazon EC2 and interact with data in different AWS data stores, for example, Amazon S3 and Amazon Dynamo DB. Amazon EMR safely and dependably handles a wide spectrum of big data use cases, including log analysis, web

GitHub Link: **https://github.com/enamshah09/DeltaComparisonOfContainers.git**

indexing, data transformation (ETL), machine learning, Cost analysis, Scientific simulation and bioinformatics.

B. Google DataProc

Google Cloud DataProc is an Apache Hadoop, which is utilized to effectively process enormous datasets requiring little to no effort with low cost. The cost can be controlled by rapidly creating managed clusters irrespective of size and turning them off when done. Cloud DataProc gives a powerful and complete data processing platform by integrating across Google Cloud Platform products.

As Containers have many advantages over hypervisor based virtualized system, its popularity in the market has gradually increased and there are many providers out in the market who are offering container based virtualized systems, we would compare different providers and analyze and come up with a most efficient and more reliable container provider. As we know the efficiency and cost of the containers, consumers would be able to take better decision which are suitable for them.

This paper is organized as follows: Section II provides the Problem analysis and Proposed Solution; Section III presents the Implementation and Evaluation; Section IV discusses the related work. The Conclusion and Limitations are presented in Section V and VI respectively. Future Work is discussed in section VII and the References are cited in the section VIII.

II. PROBLEM ANALYSIS AND PROPOSED SOLUTION

As we know that container based virtualization is at operating system level therefore provide many additional features as we discussed earlier. In this model, there are still few questions about security and isolation. In operating system level virtualization, there is no complete isolation between two virtual machines or two different users. Since container provides many other advantages as they share the same operating system kernel and manage all the process and threads in a better way. Here we are comparing two different container providers i.e. Amazon Aws, Google Cloud. As container based technology is evolved and there are many providers therefore comparing and evaluating is very important to analyze efficiency and cost of these different containers.

A. Experimental Test Bed

We are comparing the container in 2 approaches:

1. On single Node
2. On multi cluster Node

Below table shows the configuration details about single node and multi node cluster.

We have executed each benchmark three times and taken the average of 3 values and plotted the graph. We have calculated the Error Factor ((max - min)/2) which is uncertainty. To test different components of hadoop and the system as whole we have used different hadoop benchmarks.

Resource	Platform		
	AWS	*Google*	*Mac*
CPU	4 CPU/2.4 GHz, Intel Xeon Family	4 CPU/2.2 GHz Intel Xeon E5 v4	4 CPU/2.5 GHz Intel I7
Memory(RAM)	16 GB	15 GB	8 GB
Storage	50 GB	100 GB	100 GB
Operating System	Linux	Linux	Mac os

Table 1: SINGLE NODE CONTAINER CONFIGURATIONS

Resource	Platform	
	AWS EMR	*Google Data Proc*
CPU	4 CPU/2.4 GHz, Intel Xeon Family	4 CPU/2.2 GHz Intel Xeon E5 v4
Memory(RAM)	16 GB	15 GB
Storage	80 GB	100 GB

Table 2: MULTI-NODE CLUSTER MASTER CONFIGURATIONS

Resource	Platform	
	AWS EMR	*Google Data Proc*
CPU	4 CPU/2.4 GHz, Intel Xeon Family	2 CPU/2.2 GHz Intel Xeon E5 v4
Memory(RAM)	16 GB	15 GB
Storage	80 GB	100 GB

Table 3: MULTI-NODE CLUSTER SLAVE CONFIGURATIONS

GitHub Link: **https://github.com/enamshah09/DeltaComparisonOfContainers.git**

B. BENCHMARKS

The approach we are taking to solve this problem is conducting experiments on two well - known evaluation perspectives:

- Micro Benchmarks
- Macro Benchmarks

Micro-benchmark is used to test the basic components of the Hadoop system; in macro- benchmark [1], we stress the MR model and HDFS layer to test the whole system. The Hadoop distribution comes with several benchmarks, which are bundled in hadoop-*test*.jar and Hadoop-*examples*.jar. Micro benchmarks falls under test.jar and Macro Benchmark falls under hadoop-*examples*.jar

III. IMPLEMENTATION AND EVALUATION

To analyze the behavior of MR model system and how container perform, we set up a Hadoop cluster including the HDFS and the MR scheduler on single node and multi node cluster.

A. Micro Benchmarks

1) TestDFSIO: The TestDFSIO benchmarks, measures the distributed I/O performance of the application, by performing read and write test for HDFS. This benchmark is useful for tasks such as testing HDFS under stress, to identify bottlenecks in our network, to shake out the hardware, OS and Hadoop setup of your cluster machines (particularly the NameNodes and the DataNodes) and to find out how fast our cluster is in terms of I/O. TestDFSIO benchmark provides us with two notable metrics i.e. Throughput (mb/sec), Average IO rate (mb/sec). These two metrics are based on the file size written or read by each map tasks and the time elapsed to do the task. HDFS replication factor plays an important role in deciding the result of TestDFSIO.

$$\text{Throughput (N)} = \frac{\sum_{i=0}^{N} filesize_i}{\sum_{i=0}^{N} time_i}$$

$$\text{Average IO rate (N)} = \frac{\sum_{i=0}^{N} rate_i}{N} = \frac{\sum_{i=0}^{N} \frac{filesize_i}{time_i}}{N}$$

The metrics are based on writing and reading of individual files. As the size of the files have great impact on the results we took two datasets out of which one was 4GB and the other was exactly double of the first dataset i.e. 8 GB. From the results, we can deduce that higher the throughput (Mb/sec) the better is the performance. To achieve better approximation of error rate and results we performed the experiments multiple times and gradually increased the data size. We observed that after 8GB the throughput(Mb/sec) decreases linearly. From the graphs we can see that, there is not much difference in the performance of AWS and Google cloud. The native machine performance is much better than AWS and Google cloud due to less overhead.

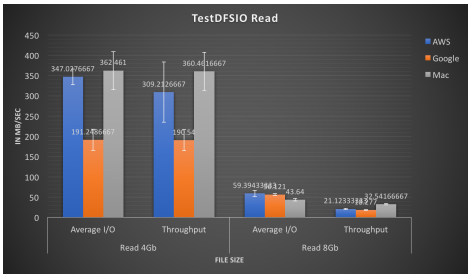

Figure 1: TestDFSIO Read Test

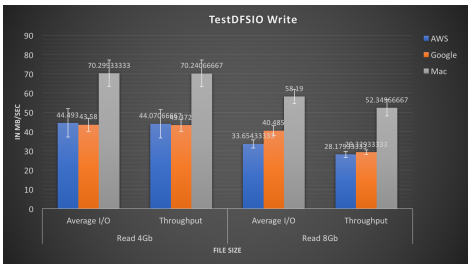

Figure 2: TestDFSIO Write Test

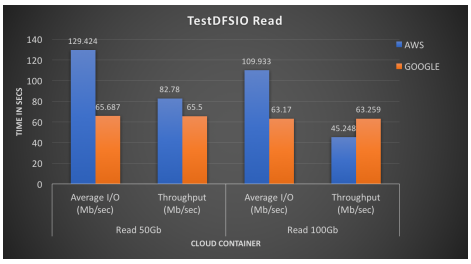

Figure 3: Cluster TestDFSIO Read Test

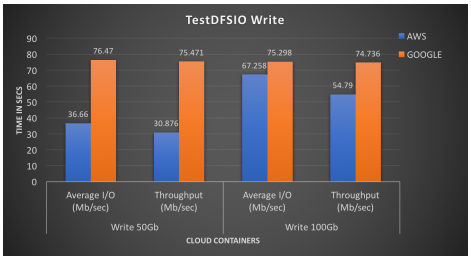

Figure 4: Cluster TestDFSIO Write Test

GitHub Link: **https://github.com/enamshah09/DeltaComparisonOfContainers.git**

2) NameNode Evaluation: NameNode(NN) Benchmark is used for the load testing the name node component of Hadoop Distributed File System(HDFS). The NameNode is tested by putting a high HDFS management stress on the NameNode. This HDFS management stress is created by generating a lot of HDFS – related requests such as creating, reading, renaming and deleting files on HDFS.As we know that NameNode maintains information about the all the files of HDFS and the locations of the files across the DataNodes, therefore NN benchmark puts load on the DataNodes and HDFS for testing the NameNode.
We performed two types of simulation as follows;

1. First, we did 'create and write' operation on the file and
2. Then we did 'open and read' operation on the file.

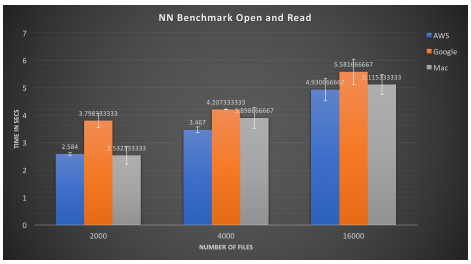

Figure 5: NN Benchmark Open and Read Test

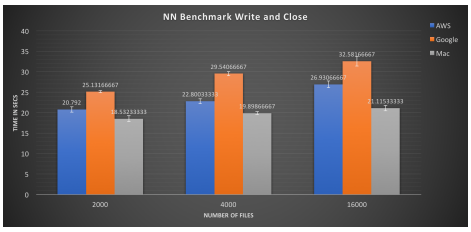

Figure 6: NN Benchmark Write and Close Test

In this benchmark, we are using 12 maps and 6 reduce functions and we generated operations on 1000 files on HDFS. In order to stress the NameNode we are varying the replication factor from 2 to 16. From the above bar charts we can observe that all the system were able to respond despite of high stress on NameNode and the performance of each of them is merely similar to native machine.

3) MapReduce Benchmark: This benchmark stresses the MR layer to identify how efficiently it works while dealing with enormous job requests. The performance of MapReduce cluster is highly influenced by the capability of job scheduler to handle huge amount of request effectively and ensuring fairness among all the jobs, as such it comes with the Hadoop distribution and runs a job multiple times, taking an average of all runs. It puts its focus on the MapReduce layer as its impact on HDFS is limited. MRBench generates the text file and reads the line as input and implements some function on it and again writes data to the file as text. The reduce function writes the value to the output again without considering the key.

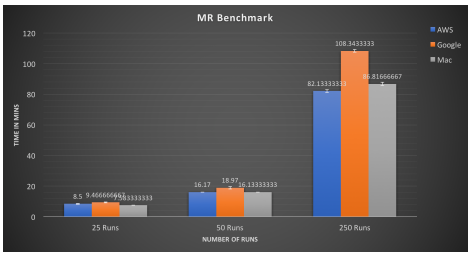

Figure 7: MR Benchmark Test

We implemented the MR Benchmark by using sequence of 25 to 250 job requests and we observed that there was no timeout and all the system could respond to the requests. Based on our configuration of system and number of runs of MR layer we saw gradually increases in time in Google when compared to AWS and native machine.

B. Macro Benchmark

4) TeraSort Benchmark: TeraSort benchmark is a most well-known Hadoop benchmark which is used to identify how fast a Hadoop cluster is. The main goal of TeraSort is to sort the 1TB data as fast as possible. It is the benchmark which tests both the HDFS and MapReduce layers of a Hadoop Cluster. A full TeraSort benchmark run consists of the following three steps:

1. Generating the random input data via TeraGen.
2. Running the actual TeraSort which uses Mapreduce on the input data
3. Validating the output data is sorted or not via TeraValidate.

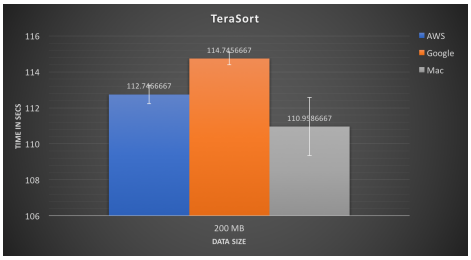

Figure 8: TeraSort Test

GitHub Link: **https://github.com/enamshah09/DeltaComparisonOfContainers.git**

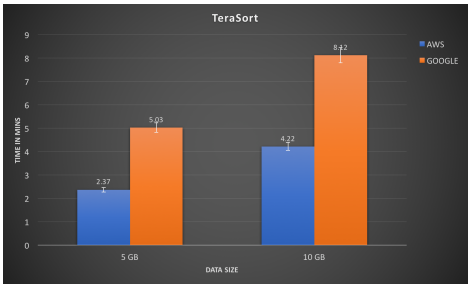

Figure 9: Cluster TeraSort Test

This test was performed on single node and multi node cluster. We used TeraGen to generate 200 Mb for single node and 5GB/10GB for multi node cluster of random data which was used as input to the TeraSort operation. After the TeraSort operation was performed, we used TeraValidate to validate the data. If any problem was detected during validation, the operation would output the keys of the data which are not sorted. The test we performed does not stress a component of cluster instead it stresses the whole cluster to analyze how the system will perform in real world applications. In single node, we can observe that in system the performance of all three did not vary a lot.

5) Analyzing performance with Wordcount: Macro-benchmarks are considered benchmarks that stress out numerous segments of a framework and can typically give more critical outcomes, as it certainly incorporates the segments before assessed by small scale benchmarks.
Word Count is a simple application that counts the number of occurrences of each word in each input dataset. It also packaged with the Hadoop bundle and it is the most widely used application to compare the performance among various Hadoop clusters. This experiment does not make any stress test, instead it only demonstrates a performance comparison when a real-world application is running on.

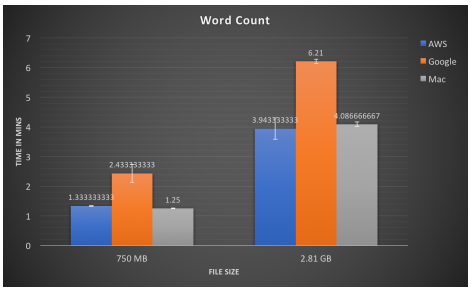

Figure 10: Word Count Test

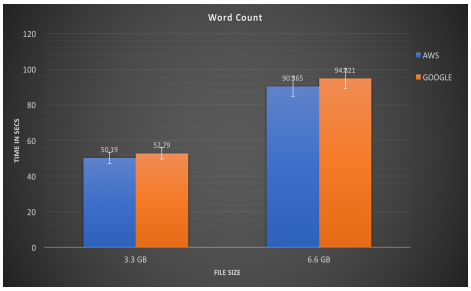

Figure 11: Cluster Word Count Test

This test was performed on single node cluster. The dataset which we used was taken from wikidata with size of 750MB and 2.8GB to run word count on single node and for multi node cluster the dataset was around 3.3GB and 6.6GB. Similar to TeraSort, we don't stress any single component rather test the system as whole in word count.
In single node, we observe that there is only slight variation in the result, we can see that when compared to google there is slight better performance in AWS. The same can be noticed even in multi node cluster.

IV. RELATED WORK

A. A Performance Comparison of Container-based Virtualization Systems for MR Clusters

In this paper, Virtualization systems is used in environments which are resource-intensive and it has been studied in the past few years because it has the potential to improve the manageability and provide better results by reducing overheads. Nevertheless, the results obtained so far have proved that virtualization system is still not able to achieve the same native performance [1]. The author also states that although the trend is towards the usage of MR clusters for better performance there are still no studies on evaluating the overhead caused due to containers. To evaluate the performance, they have performed several experiments using different benchmarks first to stress the components of Hadoop cluster and then the cluster as whole.

B. Performance Analysis of Container-based Hadoop Cluster: OpenVZ and LXC

The paper focuses on container - based virtualization implementation on virtualized Hadoop cluster and performance analysis of container-based Hadoop cluster on OpenVZ and LXC by its I/O throughput and average execution time on Hadoop MapReduce. They are using container virtualized cluster for measuring the performance in terms of I/O performance overhead, and average execution time of MapReduce job. Several benchmarks are used for measuring the performance [2]. Hadoop packages provide different applications such as word count, sorting huge amount of data on which we run different benchmarks. After analyzing different benchmark results, they found out OpenVZ is more stable and provides more better results than LXC for the system configuration they used.

GitHub Link: **https://github.com/enamshah09/DeltaComparisonOfContainers.git**

V. CONCLUSION

This paper presented a performance comparison between the different cloud providers Amazon Aws and Google Cloud and even compared those results with physical system for single node cluster. We also compared AWS EMR and Google DataProc for multi node cluster.

Virtualization plays an important role in terms of security, resource utilization and better server management. The capacity of cloud computing environment in supporting MapReduce workload have developed in the most recent years due of the expansion volume of information we confront today and the high cost for figuring such an extensive dataset in cluster groups. Different users or institutes which may have distinct software prerequisites that vary on another share the traditional MapReduce cluster. Mesos and Yarn are frameworks that take advantages from utilizing a container based framework to give better resource sharing between software systems, for example Hadoop and MPI.

As we know that configurations changes from each platform it's difficult to compare only using parameters so we wanted to compare cost vs efficiency

Single Node Cluster (AWS EC2 vs Google)

Total Cost = Time x R x cost/hour
R = number of times the task is repeated

We took Word Count benchmark to compare cost vs efficiency because it is a macro benchmark, therefore it tests the whole system.

As per our result, Word Count benchmark for 2.81 GB in Amazon got completed in 3.94 mins and if the task is repeatedly run for 480 times, then the total time for running the benchmark is 31.544 hours and the cost will be $ 0.188 x 31.544 = **$5.93**

For Google, Word Count benchmark for 2.81 GB got completed in 6.21 mins and if the task is repeatedly run for 480 times, then the total time for running the benchmark is 49.68 and the cost will be $0.134 x 49.68 = **$6.657**

From the above results and on the basis our analysis we can predict that Amazon performs better as compared to google in terms of cost.

Multi Node Cluster (AWS EMR vs Google DataProc)

Total Cost = Time x R x cost/hr;
R = number of times the task is repeated.

As TeraSort is a macro benchmark, it tests the whole system. Therefore, we took TeraSort benchmark to compare cost vs efficiency.

As per our result TeraSort benchmark for 10GB in Amazon is completed in 4.22 mins and if the task is repeatedly run for 480 time, then the total time for running the benchmark is 33.76 hours and the cost will be $0.21 x 33.76 = 7.09.

As per our result TerasSort benchmark for 10GB in Google is completed in 8.42 mins and if the task is repeatedly run for 480 times the total time is 67.36 hours and the cost will be $0.12 x 67.36 = 8.08.

From the above results and on the basis our analysis we can predict that Amazon performs better as compared to google in terms of cost.

VI. LIMITATION

1. The physical machine we are testing has different CPU processor (I7 intel) as compared to cloud providers (intel Xeon).
2. Cloud providers does not provide details about network bandwidth.
3. To test few benchmarks like TeraSort we need to generate 1TB of data, to test the full capabilities of a multi node cluster due to resource constraints we not able to generate so much of data.
4. In single node cluster, NameNode enters safe mode when there is shortage of memory. As a result, the HDFS becomes readable only. That means one cannot create any additional directory or file in the HDFS. The amount of data NameNode can store is limiting factor to test NN benchmark.

VII. FUTURE WORK

As we have limited resource we were not able to fully test the capabilities of the cloud providers. In our Future work, we would even consider storage space cost and how many containers are required for specific workload and how the network performance vary as cluster size increases with simultaneously increase in replication factor. As we know that few configuration changes depending on the regions we would even give specific region comparisons to more accurate results.

If we have deep insight all the factors which influence the container technology from this data, we can recommend consumer or any firms to which cloud provider is best suited for them. The comparison techniques used can be further extend to any cloud provider in the market like Microsoft AZURE, Rackspace etc.

VIII. REFERENCES

[1]. M. G. Xavier; M. V. Neves; C. A. F. D. Rose, "A Performance Comparison of Container-Based Virtualization Systems for MapReduce Clusters", 2014 22nd Euromicro International Conference on Parallel, Distributed, and Network-Based Processing, pp 299 – 306.

[2]. Rizki Rizki, Andrian Rakhmatsyah and M. Arief Nugroho, "Performance Analysis of Container-based Hadoop Cluster: OpenVZ and LXC", 2016 4th International Conference on Information and Communication Technology (ICoICT), pp. 1 – 4.

[3]. Hadoop, "Apache Hadoop Website," 2013, accessed on August 2013. [Online]. Available: http://hadoop.apache.org

[4]. D. Borthakur, J. Gray, J. S. Sarma, K. Muthukkaruppan, N. Spiegelberg, H. Kuang, K. Ranganathan, D. Molkov, A. Menon, S. Rash et al., "Apache hadoop goes realtime at facebook," in Proceedings of the 2011 international conference on Management of data. ACM, 2011, pp. 1071–1080.

[5]. "Amazon Elastic MapReduce (Amazon EMR)," 2013. [Online] Available: http://aws.amazon.com/elasticmapreduc

[6]. A. Kivity, Y. Kamay, D. Laor, U. Lublin, and A. Liguori, "kvm: the linux virtual machine monitor," in Proceedings of the Linux Symposium, vol.,

GitHub Link: **https://github.com/enamshah09/DeltaComparisonOfContainers.git**

2007, pp. 225–230. [5]

[7]. P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield, “Xen and the art of virtualization,” ACM SIGOPS Operating Systems Review, vol. 37, no. 5, pp. 164–177, 2003

Comment [b1]: Can u fit in the above page

GitHub Link: **https://github.com/enamshah09/DeltaComparisonOfContainers.git**