International Journal of Engineering & Technology, 7(4.16) (2018) 28-31

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET

Research paper

Enhancing Virtual Machine Live Migration Time Using Vcpu Limits

Aula Abdel Latief Dewan [1]*, Rabah Abood Ahmed [2]

[1] *Al-Nahrain University, College of Information Engineering*
[2] *Al-Nahrain University, College of Information Engineering*
**Corresponding author E-mail: [1] aula.aldewan@gmail.com, [2] rabahalansary@coie-nahrain.edu.iq*

Abstract

In the Cloud Computing, the live migration of a virtual machine or VM from one physical machine to another is a vital process applied on the service provider side. Live migration helps administrators manage data centers resources optimally. Due to the intensive daily use, it is necessary to improve the performance of VM migration-this is reflected in improving the quality of service provided to the customer while minimizing the costs incurred. Pre-copy is an important method of live migration that has been adopted in many cloud computing platforms. One main drawback of this method is the degradation of its performance when the migrating VM runs one or more of the processes that write on the memory pages faster than the speed of transferring those pages. This makes migration time-consuming. In this paper, we propose an approach to address this issue by reducing the generation rate of the modified pages while maintaining the service provided to the customer. This approach was applied to the real Xen platform and the results showed an improvement in the time of live migration of the virtual machine that runs an intensive write process up to 40% compared to the migration time of the original Xen platform.

Keywords:. *virtual machine, live migration, pre-copy method, virtualization, cpulimit, cloud computing.*

1. Introduction

Cloud computing is popular in today's terminology, provides on-demand hosting services to organizations and users with fast and reliable computing model using large data centers [1]. In general, cloud computing contains three models. Software as a Service (SaaS), which provides software as a service to the client without having to be installed on client machine, such as Google Docs. Platform as a service (PaaS), which provides a complete platform for the developer. The third model is Infrastructure as a Service (IaaS) model, pay for using computing hardware resources rather than owning them. Normally, Cloud competing provide IaaS to clients via virtual machines (VMs). On the provider side, a large number of VMs need to be migrated every day from one server to another. This VM migration process is required for server management operations (update, upgrading, maintenance, etc.), also, for network load balancing and saving energy. To ensure minimal impact on VMs availability and performance, provider operations are performed via live migration: the process of relocating the running VM from one server to another with a negligible downtime under normal conditions.

Pre-copy is a method for live VM migration. This method is used widely in various cloud computing approaches [2]. Pre-copy is an iterative method where memory pages are transferred in iterations. In pre-copying, the first round is to transfer all memory, CPU, and disk information to the destination server. During this transfer period, some memory pages can be changed (modified) and must be moved in the succeeding iteration. This scenario can be repeated when migrating a VM that is running a process that intensively modifies memory pages. In this paper, we will refer to such a process by IMMP. Running the IMMP process can make the rate of modified pages become higher than the rate of transmitted pages. Consequently, the number of iterations may increase. When the allowed number of iterations is exceeded the operating system will interrupt the live migration operation to starts the stop and copy phase. This problem can incur the live migration operation more time to complete.

Several approaches have been introduced in the literature to alleviate this problem. These approaches can be classified into two types. The first is to increase the transmission rate of memory pages either by reducing the amount of memory size need to be transfer via flushing the unused memory [3], or removing duplicate pages [4], or delay transfer the modified pages that have a low frequency of use [5]. The second type is via controlling the IMMP processing. Jin et al. [6] suggested limiting the CPU usage for the virtual machine being migrated when it is running one or more IMMP processes. This method schedules the CPU time for this virtual machine to the required percentage to reduce the frequency of the modified pages and improve the pre-copy migration process to the accepted level. This method was proposed as the last choice. It is difficult to determine CPU processing time for all applications running on the VM without compromising its performance to be unacceptable to clients. Clark et al. [7] suggested the Stunning Rogue Process technique to alleviate this problem within the migrated operating system. The idea is to create a thread within the guest operating system that monitors the write behavior of each individual process. This thread imposes a delay to any IMMP process violates the specified number of writes on memory pages by moving it to a waiting queue.
In this paper, we will introduce a new technique combining both the aforementioned techniques [6], [7]. The strategy is not to con-

trol the use of vCPU outside the VM as in [6] but within the VM to reduce the processing time of IMMP through the guest operating system. Different interactive services are run during VM migration time, so it is desirable to determine the best vCPU limit without compromising the quality of service provided to customers.

The rest of the paper is organized as follows. The description of the current techniques of VM migration will be presented in Section II. The idea of the proposed approach is presented in Section III. Section IV will describe the evaluation methodology of the proposed approach, while the results will be discussed in Section V. The conclusion of the paper is given in Section VI.

2. VM Migration Techniques

The migration of a virtual machine in the cloud environment can be divided into two types: non-live migration and live migration. For the non-live migration, the operating system stores a checkpoint for the current state of the virtual machine and completely stops the VM at the source device and then starts the migration operation. After transferring all of the memory pages, CPU state, and disk data to the destination device, the operating system will restore the checkpoint and starts the virtual machine from the last saved state. During migration time the virtual machine will be inaccessible by the client. The advantages of this type are the simplicity of implementation and its transferring of memory pages all at once, not by iterations.

As for the live migration, there are post-copy migration and pre-copy migration. Post-copy suspends the migrating virtual machine at the source node, begins to transfer required CPU state to the destination node, resumes the virtual machine, the rest of memory pages and CPU statues will be transferred on demand [2]. Although the Post-copy can reduce migration time and guarantees transfer memory pages only once, however, the increase of memory faults can cause more network service disturbance [8].

Pre-copy migration is a set of iterations used to perform the live migration. The first iteration transfers all memory pages, CPU states and disk data from the source node to the destination node. The succeeding iteration is used to move back the memory pages that were changed during the previous iteration. These memory retransfer iterations continue until either the allowed number of iterations is completed, or the number of modified pages in the current iteration is less than a threshold number allowed by the pre-copy. The last iteration of the pre-copy is the stop-and-copy, which suspends the virtual machine at the source node, transfers all memory pages, CPU states of the currently running processes, network data, etc to the destination node, then resumes the virtual machine. Using pre-copy can robust the migration process by retrieving it from the source node in case of migration fails. But, the migration performance may adversely be affected when running one or more IMMP processes during migration. The negativity of the pre-copy in handling the retransferred iterations of the modified pages may cause to fail the live migration process. In this paper, we propose an approach to mitigate this problem by controlling the rate of the modified pages generated by IMMP during migration.

3. Proposed Approach

This work presents a technique to improve the pre-copy method. This technique could reduce the number of iterations, and thus improves the live migration performance.

The main idea of this approach is, when the VM executes IMMP process during migration, it limits the vCPU percentage for this IMMP only, so that the modified pages rate deceases to a value small enough to reduce the total live migration time, while maintaining IMMP performance to be acceptable to customers.

4. Evaluation

This section describes the experimental environment for evaluating the migration and performance metrics of the proposed approach and compares them with the basic method of pre-copying.

4.1 Experimental Setup

In order to test this approach, two identical computers with AMD Quadro Core 2.70 GHz processors, disk capacity of 200 GB, and 4.00 GB memory were used as source and destination servers. In addition, a laptop with i5 Intel®core was used as a client node. Mikrotik 260GS switch with 100 Mbps was used to connect the three machines in star topology local area network. The migration process was performed on a virtual machine with one vCPU, a disk capacity of 4.6 GB and a memory size of 490.7 MB. This virtual machine is stored directly on the local storage of the source server; no storage area network was used during the experiments. For operating systems, VM used Linux Ubuntu 12.04 LTS, and for virtualization, we used Xen Server 6.5 as a hypervisor in both servers.

ye et al [9] have demonstrated that the total migration time and downtime of the VM migration process are heavily influenced by memory usage, and "compress, startup, and mpeg audio workloads are memory-intensive and incur rapid pollution of memory pages, which eventually affects the total migration time and downtime." Therefore, as a workload, we used HD MP4 streaming video files over the Internet provided by local network service provider. This choice is for experiencing the worst case scenario, where we found that HD MP4 stream files generate more modified memory pages than mpeg audio streaming files.

To restrict available vCPU usage, the CPU limit application was used [10]. This open-source Linux tool provides the ability to manage CPU utilization for a particular process without significantly affecting overall system performance. To evaluate the proposed approach, many experiments were conducted on VM migration for different percents of vCPU usage. Each experiment is performed with a different workload, as the MP4 video process continues throughout the migration process.

4.2 Quality of Service (QoS)

In this study, two metrics are used to evaluate the QoS for the proposed approach, which are migration metrics and performance metrics [11]. These evaluation metrics are evaluated from a single migration perspective. In the migration metrics two measuring factors are considered, VM migration time and downtime. VM migration time represents the total time required to migrate the VM from source to destination. Whereas downtime is represents the period of time that the VM is not reachable for client requests during the VM migration time. The ping operation is used to determine the downtime which is equal to the time of packet loss.

In the performance metric, the degradation in video streaming is used to evaluate the impact of limiting vCPU usage on workload performance. This evaluation determines the ability of the VM to provide an acceptable video service with non-stop streaming.

5. Results and Discussions

The experimental environment described in subsection 4.1 has been used to compare the proposed approach to Xen pre-copy method. During the experiments, the IMMP process that represents the workload is predetermined, and the CPUlimit tool is utilized by the guest operating system to control the percentage of vCPU usage for this IMMP process.

In migration experiments and for VM migration time factor, the results revealed a significant reduction in VM migration time when we implement the proposed approach as shown in Fig.1. To determine the minimum time to migrate the virtual machine, ten live migrations were performed on VM machines that did not execute any workload (inactive state) using the basic Xen pre-copy method and the average was calculated. Fig.1 shows the results of using the proposed approach as an increment ratio in VM live migration time to the minimum migration time. These results indicate that the introduction of vCPUlimit to control the workload process will improve VM migration time. This improvement is occurred by reducing the subsequently modified pages to reach the threshold point to stop the pre-copying phase and to start the stop and copy phase. This threshold point is reached when the modified pages are very small and are generated very much faster than the copying operation, to prevent waste of resources and time. This improvement is reflected as a decrease in VM migration time compared to the basic migration of the Xen server, which results in a 61.7% increase in the minimum migration time. Reducing vCPU limits to 10% improves the migration time to near the minimum time, however, VM performance is severely affected as will be described shortly. The best scenario was determined by reducing the vCPU to 15%, 20% and 25% of use, with improvement reaching 38%, 40% and 36%, respectively, compared to the original Xen server method.

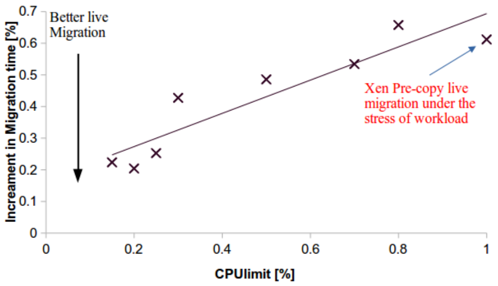

Fig. 1: CPU usage reduction for a memory-intensive process and its effects on the time consumed by VM live migration

As for VM downtime, the results show that the proposed approach will not reduce VM downtime. Fig. 2 illustrates the results obtained from the experiments, which show, however, an increase in VM downtime compared to the basic method of Xen (pointed as No vCPUlimit). The downtime in Xen pre-copy depends on several factors, one of which is the size of the last iteration data that must be transferred at the stop-and-copy phase. The more time it takes to transfer last iteration data, the longer the time required at the stop-and-copy mode where the virtual machine should suspend. Also, as it has been observed by [7], decreasing downtime increases with increasing the number of additional iterations. Fig. 3 shows the rates of modified pages in memory during the live migrating when using the proposed approach of 15% and 25% vCPUlimit compared to the basic Xen method with and without workload. Thus, we can explain the increase in VM downtime when we use the proposed approach is because of the low number of iterations. Reducing the modified pages induced by the IMMP is not as effective as increasing iterations in reducing VM downtime. Note that the latest round data is not just about modified pages caused by the IMMP process. This data also contains network and disk pages, as well as, the pages of the stack and local variables for the executing processes [7].

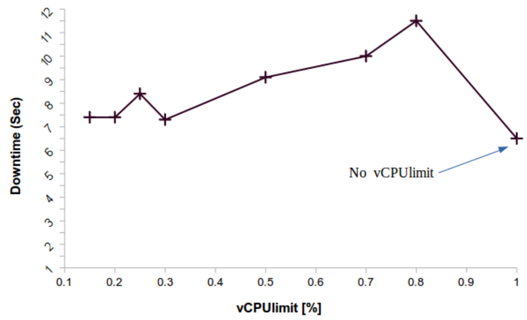

Fig. 2: The downtime results of introducing the proposed approach and Xen original pre-copy method

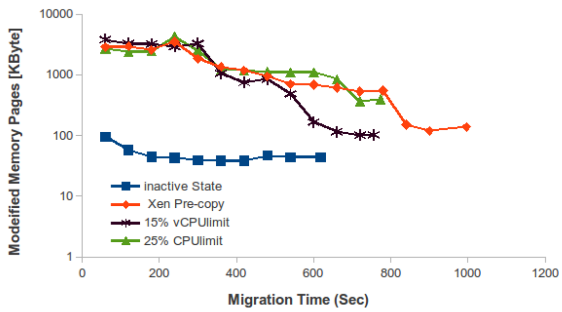

Fig. 3: Modified pages trends in memory during VM live migration using vCPU limits 15%, 25% compared to the basic Xen method with and without workload

In the performance metric, Fig.4 depicts the effects of reducing vCPU usage on video streaming performance. Pro-viding only 10% of vCPU time for processing the MP4 video workloads degrades the video streaming performance to unacceptable limits. The performance of streaming reduced to about 30% of video operation is non-stop streaming. To determine the accepted performance, we conduct several experiments for each vCPU usage limits, in step of ten percent, as shown in Fig.4. These experiments show that increasing vCPU processing limits to 30% is sufficient for accepted video streaming performance. To determine the point of minimum vCPU limits with accepted performance, we focus on the range 15% - 25%. The embedded figure in Fig. 4 shows that a 25% of vCPU is sufficient for accepted video streaming performance. So far, despite the limited increase in VM downtime, the experiments reveal that as much as we can reduce vCPU usage for IMMP processes as much as we can reduce VM migration time. This reduction can not be less than the migration time of the inactive VM (the minimum migration time). Limiting vCPU usage for an IMMP process to 25% can be sufficient to provide accepted service to the customer.

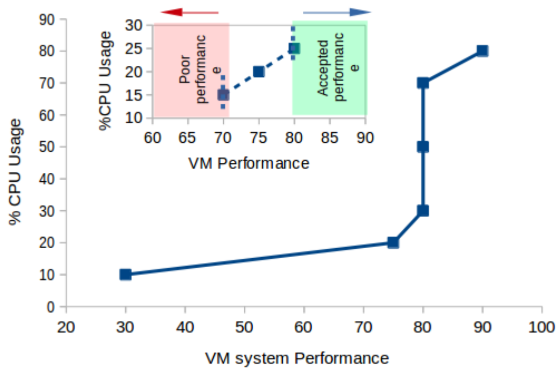

Fig. 4: Workload performance degradation

6. Conclusions

In this paper, we proposed an approach to improve the live migration performance of virtual machines based on the pre-copy method. In pre-copying, the migration performance is sensitive to the rate of modified pages in the virtual machine memory during migration. The idea was to limit the vCPU usage for the process that polluting memory pages. Comparison results with the original pre-copy method showed an improvement in migration performance due to migration time while maintaining customer service and virtual machine downtime.

References

[1] N. Kshetri, "Cloud computing in developing economies," Computer, vol. 43, no. 10, pp. 47–55, 2010.

[2] M. R. Hines and K. Gopalan, "Post-copy based live virtual machine migration using adaptive pre-paging and dynamic self-ballooning," in Proceedings of the 2009 ACM SIGPLAN/SIGOPS international conference on Virtual execution environments. ACM, 2009, pp. 51–60.

[3] C. A. Waldspurger, "Memory resource management in vmware esx server," ACM SIGOPS Operating Systems Review, vol. 36, no. SI, pp. 181–194, 2002.

[4] C. P. Sapuntzakis, R. Chandra, B. Pfaff, J. Chow, M. S. Lam, and M. Rosenblum, "Optimizing the migration of virtual computers," ACM SIGOPS Operating Systems Review, vol. 36, no. SI, pp. 377–390, 2002.

[5] H. Liang, H. Dai, X. Pei, and Q. Zhang, "A new pre-copy strategy for live migration of virtual machines," in Identification, Information and Knowledge in the Internet of Things (IIKI), 2016 International Conference on. IEEE, 2016, pp. 54–59.

[6] H. Jin, W. Gao, S. Wu, X. Shi, X. Wu, and F. Zhou, "Optimizing the live migration of virtual machine by cpu scheduling," vol. 34, pp. 1088–1096, 07 2011.

[7] C. Clark, K. Fraser, S. Hand, J. G. Hansen, E. Jul, C. Limpach, I. Pratt, and A. Warfield, "Live migration of virtual machines," in Proceedings of the 2nd Conference on Symposium on Networked Systems Design & Implementation-Volume 2. USENIX Association, 2005, pp. 273–286.

[8] R. Boutaba, Q. Zhang, and M. F. Zhani, "Virtual machine migration in cloud computing environments: Benefits, challenges, and approaches," in Communication Infrastructures for Cloud Computing. IGI Global, 2014, pp. 383–408.

[9] K. Ye, X. Jiang, D. Huang, J. Chen, and B. Wang, "Live migration of multiple virtual machines with resource reservation in cloud computing environments," in Cloud Computing (CLOUD), 2011 IEEE International Conference on. IEEE, 2011, pp. 267–274.

[10] CPUlimit - CPU Usage Limiter for Linux:, Sept. 15 2007. [Online]. Available: http://CPUlimit.sourceforge.net.

[11] D. Li, W. Wang, Q. Li, and J. Cheng, "A comprehensive evaluation of scheduling methods of virtual machine migration for energy conservation," IEEE Systems Journal, vol. 11, no. 2, pp. 898–909, 2017.