Computer Engineering and Applications Vol. 8, No. 2, June 2019

An Approach to Improve the Live Migration Using Asynchronized Cache and Prioritized IP Packets

Mohammad Reza Moslehi Takantapeh[1], Keyvan Mohebbi[2,*]

[1]*Department of Computer Engineering, Isfahan (Khorasgan) Branch, Islamic Azad University, Isfahan, Iran*
[2]*Department of Electrical and Computer Engineering, Mobarakeh Branch, Islamic Azad University, Mobarakeh, Isfahan, Iran*
**Correspondence: k.mohebbi@mau.ac.ir*

ABSTRACT

The live migration of a virtual machine is a method of moving virtual machines across hosts within a virtualized data center. Two main parameters should be considered for evaluation of live migration; total duration, and downtime of migration. This paper focuses on optimization of live migration in Xen environment where memory pages are dirtied rapidly. An approach is proposed to manage dirty pages during migration in the cache and prioritize the packets at the network level. According to the evaluations, when the system is under heavy workload or it is running within a stress tool, the virtual machines are intensively writing. The proposed approach outperforms the default method in terms of number of transferred pages, total migration time, and downtime. Experimental results showed that by increasing workload, the proposed approach reduced the number of sent pages by 47.4%, total migration time by 10%, and the downtime by 27.7% in live migration.

Keywords: Live Migration, Xen, Virtual Machine, Dirty Page, Downtime

1. INTRODUCTION

Clouding and virtualization have become a core facility in modern computing installations. This provides opportunities for improved efficiency by increasing hardware utilization and application isolation as well as simplifying resource allocation and management. The widespread use of cloud computing in road networks is taken for granted as well [1]. Availability and reliability in location based services (LBSs) applications allow vehicles to quickly exchange information in order to prevent accidents and hazardous situations [2, 3]. In the road networks and many other sensitive places, information is constantly and periodically broadcasted and sustainable infrastructure for this kind of communication is essential [4]. Hence, along with improving protocols and methods to increase security and efficiency in the road networks, we need some approaches for increasing stability in the datacenters as well [5-7].

One of the key features in the stability of virtualization and cloud computing is live migration. Live migration platforms allow administrators to move running virtual machines (VMs) seamlessly between physical hosts. This is of particular benefit for service providers hosting high availability applications.

Policies such as 99.999% availability in the telecommunication industry permit only 5 minutes of downtime per year. Routine activities such as restarting a machine for hardware maintenance are very difficult under such a condition. Hence, service providers invest a great deal of resources in high-availability and fault-tolerant systems [1].

The word “live”, points to the fact that the migration should be transparent to the users. Therefore, the guest operating system should be running during migration. The downtime is the time it takes for the source host to suspend execution of the VM until the destination host resumes it. The VM should not be stopped for a considerable amount of time so that it is usable during migration and thereby transparent to the users who are not ideally aware of the occurring migration [8, 9].

Modern services often provide live migration functionality to make it easier for VMs to move without performing many other administrative works [10]. Main benefits of VM migration are power saving, load balancing and memory releasing. Fast live migration can improve global system utilization, system serviceability and availability by moving VMs [11].

Primary migration is based on cold migration in which migration of all VMs of the desired destination must be stopped and after that, machines can be resumed at the destination. Delay in sensitive applications is the main problem of the provided method; therefore, live migration for the sake of reducing the latency of migration has been proposed [1, 12]. The services running during live migration trend continue performing and are suspended only in a short period. There are two important criteria in the live migration; the total amount of live migration, and the downtime of the VM with running services. The transfer time of origin to destination in this service is desired. The total migration time can be reduced by decreasing the total number of pages that is transferred from the source to the destination. On the other hand, whenever the amount of time of stopping and resuming in VM is reduced, customer satisfaction increases [13, 14].

In this study, an approach is proposed to reduce the number of dirty pages, total migration time and downtime. It manages dirty pages in the cache and prioritizes the packets at the network level.

This paper is organized as follows. The preliminaries and related works are reviewed in section 2. The proposed approach is described in section 3. The parameters affecting the proposed approach are introduced in section 4. It is then evaluated in section 5 and compared with similar approaches in section 6. Finally, the paper is concluded in section 7.

2. PRELIMINARIES AND RELATED WORKS

In the following, first some preliminaries on the virtual machine live migration are presented, then the prominent research works in this subject are reviewed.

2.1 PRELIMINARIES OF THE LIVE MIGRATION

To get the always availability of the data, there are many live migration techniques in the virtualization-based environment. The pre-copy approach is one of the live migration methods which takes minimal downtime of VM being migrated in comparison of the others. For the duration of migration, pages of physical memory transfers from a primary host to the new host, while the state of the VM is running on the primary host. During the migration of VMs, consistency and integrity must be preserved. After that iterative procedure of the transportation dirty pages, stop-and-copy phase will be initiated and executed for a while and that caused VM

ISSN: 2252-4274 (Print)
ISSN: 2252-5459 (Online)

suspended, the remaining pages are transferred, due to this hypervisor of the destination VM generate a signal for the resumption of the executed VM [15-17].

In addition to the pre-copy procedure, there are some other related techniques that offer a better solution to optimize migration time like post-copy migration technique that has advantages over the pre-copy method. Some research works show that the downtime of post-copy migration for a VM being migrated is less than the time of the pre-copy migration. However, pre-copy procedure supports the para-virtualized users so the catching memory method can provide memory managing based on pseudo-paging system within a guest. Then, pre-copy is currently more common migration procedure [16, 17]. Hines et al. [14] compared post-copy against pre-copy approach on top of the Xen hypervisor. The results showed improvements in several migration metrics including pages transferred, total migration time and network overhead using a range of VM workloads. They used post-copy with adaptive pre-paging so that it removed all duplicate page transmissions. This approach was able to reduce the number of network-bound page faults for large workloads.

2.2 RELATED WORKS TO IMPROVE THE LIVE MIGRATION

In check point/reply technique proposed by Haikun Liu et al. [18], the logs of the source VM activities are recorded and transferred to the destination machines rather than memory pages, thereby the bandwidth consumption is reduced and migration time of the process is enhanced.

Kumar Bose et al. [12] proposed combining VM replication with VM scheduling to overcome the challenge of migration latencies associated with moving large files (VM images) over relatively low-bandwidth networks. They replicated a VM image selectively across different cloud sites, chose a replica of the VM image to be the primary copy, and propagated the incremental changes at the primary copy to the remaining replicas of the VM. The proposed architecture for integrated replication and scheduling, which is called CloudSpider, was capable of minimizing migration latencies associated with live migration of the VM images across WANs.

Svard et al. [19] implemented delta compression live migration algorithm as a modification to existing algorithms of the KVM hypervisor. They studied the application of delta compression during transfer of memory pages in order to increase migration throughput and thus reducing downtime. Its performance was evaluated by migrating VMs running different type of workloads and their evaluation demonstrated a significant decrease in migration downtime in all test cases. Using delta compression, risk of service was reduced as data was stored in the form of changes between versions.

Ibrahim et al. [20] presented a performance analysis of the current KVM implementation and studied pre-copy live migration of MPI and OpenMP scientific applications running on KVM. The main idea of this approach was to control migration based on applying memory rate of change. They presented a novel algorithm that achieved both low downtime and low application performance.

Fei Ma et al. [21] improved pre-copy approach on Xen 3.3.0 by adding a bitmap page and putting frequently updated pages into it by evaluating the iteration process. The frequently updated pages are then only transmitted in the last round of the iteration process which guarantees only one transmission.

Akoush et al. [1] showed that the production rate of dirty pages and network bandwidth had a significant impact on live migration but a non-linear effect on migration performance. Migration times were accurately predicted to enable more dynamic and intelligent placements of VMs without degrading performance.

Safari et al. [22] improved pre-copy by taking advantages of clock policy for tracking transmission of those frequently updated pages repeatedly. This could ensure that those updated pages in the working set were transmitted just once in the iteration process.

Ei Phyu Zaw et al. [23] proposed a framework that included pre-processing phase in traditional pre-copy based on live migration for reducing the amount of transferred data. Their technique comprised three phases: Pre-processing phase, Push phase, Stop and Copy phase. The Pre-processing phase transferred the last recently used pages in the working set list. The Push phase transferred the memory pages that were not in the working set list from source VM to destination VM. The Stop and Copy phase forced the processor state and working set list to migrate to the destination VM. This technique effectively reduced the number of iterations by retaining the required memory pages of the source machine until the last step of migration. Applying the proposed algorithm that combined LRU (Least Recently Used) cache with splay tree algorithm reduced the amount of transferred memory page.

3. PROPOSED APPROACH

Bombarding the disk by writing a torrent of data causes large pauses on the system. The cache then must decide how to manage the data that means an additional burden on the system. Therefore, cache is converted from asynchronous mode to synchronous mode that creates a delay in data transmission

In order to preserve asynchronous communication, instead of using one absolute parameter in the cache, two parameters are used. First parameter, called PrmA, is a percentage of total memory that includes dirty and recovered pages before writing the background kernel processes on disk starts. Second parameter, called PrmB, is the absolute amount of total system memory with dirty pages before being written on disk. Unlike previous methods, in this approach all mentioned parameters and adjustments of cache are introduced in the hypervisor level.

Flowchart of the proposed approach is depicted in Figure 1. As shown in this figure, when the "Starts Migration" flag is zero, migration is not started yet and the system performs normally. If this flag changes to one, a migration will occur. From this point, the proposed approach starts and in step 8, when all dirty pages are moved, live migration will be completed and Stop and Copy step will begin.

In step 4, after the start of live migration and setting the IP address priority to the highest value, writing the data in the cache will begin, but the data will not be transferred until PrmA reaches to the PrmA_minThreshold. After that, data transfer is started asynchronously. In step 7, PrmB is checked and if it reaches PrmB_minThreshold, asynchronous writing will be stopped and synchronous writing begins. This process continues until live migration finishes.

ISSN: 2252-4274 (Print)
ISSN: 2252-5459 (Online)

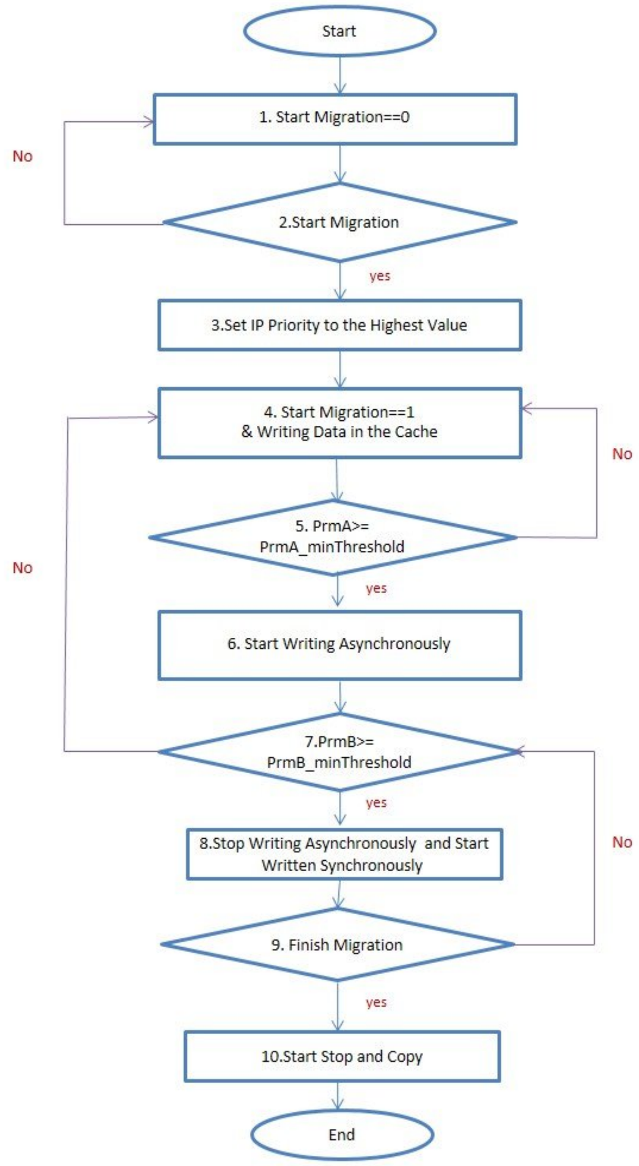

FIGURE 1. Flowchart of the proposed approach

4. PARAMETERS OF THE PROPOSED APPROACH

One of the main problems in live migration is how dirty pages are managed. Optimal management of dirty pages helps reducing migration time and number of transferred pages. In contrast to many existing works, the proposed approach tries to increase the efficiency of the system without affecting other aspects. The focus of this approach is on managing dirty pages and reducing the number of transmitted packets by operating on parameters such as cache and transmitted packets.

4.1 PRIORITIZING LIVE MIGRATION PACKETS

One of the parameters affecting management of dirty pages is live migration packets priority. By assigning higher priority to live migration packets for moving towards destination, performance can be improved and total migration time can be decreased. If we do not apply the mentioned approach, then transmitted live migration packets will not have priority over other network packets. Thus, the probability of both losing data and repeating data transmission in live migration will be increased.

The virtual machines, which will fail if there is a lack of resources like RAM and processor, are located on the servers and decide to migrate according to the considered policy. If a high amount of time is consumed for displacement in live migration, data might be lost or virtual machines might not be accessed. Thus, in the proposed approach, first all live migration packets are prioritized so that migration is performed faster and the probability of losing migration packets is reduced.

In order to apply priority, a field called Type of Service (TOS) is used in IP header. TOS field is responsible for managing datagram and prioritizing packets. In fact, this field prioritizes packets based on conditions so that data centers systems can decide and proceed accordingly [24]. In addition, prioritizing live migration packets provide the best performance when the traffic network is at its peak.

As can be seen in Figure 2, 802.1Q is a 4-byte field in IEEE 802.3 standard. This optional field is located in second layer of OSI model. By changing these 4 bytes, priority of migrating virtual machines packets can be increased. Therefore, switches decide based on priority of packets and migrating virtual machines packets receive the highest priority.

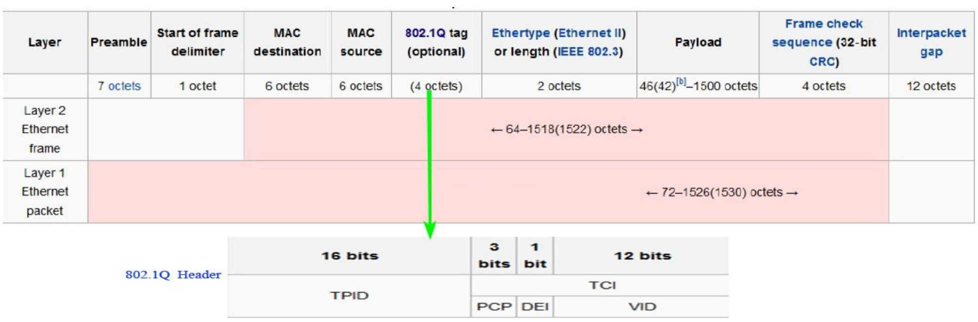

FIGURE 2. IEEE 802.3 standard and 802.1Q protocol header

As can be seen in Table 1, the highest priority of PCP, 7, is associated with network control type [24].

ISSN: 2252-4274 (Print)
ISSN: 2252-5459 (Online)

TABLE 1
Network traffic priorities

PCP	Priority	Traffic Types
1	0 (Lowest)	Background
0	1	Best Effort
2	2	Excellent Effort
3	3	Critical Applications
4	4	Video<100 ms latency and jitter
5	5	Voice< 10 ms latency and jitter
6	6	Internetwork Control
7	7 (Highest)	Network Control

In order to increase priority of TOS field, some changes must be implemented in live migration file. First, VLAN should be defined so that data centers systems can decide correctly considering changes in priority of packets. Then, by mapping TOS to PCP, which is located in the header of 802.1Q protocol, priority in second layer of OSI model can be increased. Hence, each communication has its own priority in second layer of OSI model. In other words, only live migration packets should be prior and not all the packets.

4.2 CACHE AND PRESERVE ASYNCHRONOUS COMMUNICATION

Cache is one of the effective parameters in increasing the speed of a system. Increasing or decreasing the cache size affects the system performance. However, if there is not an appropriate management, this issue can become a bottleneck [24].

Apart from that increasing cache leads to loss of data, it can also increase the risk of long I/O pauses for the system. Because the cache gets full and needs to be emptied and searching the cache may take a long time. This issue causes disaster when data writing speed is too high and disk speed is too low. Decreasing the cache in an unplanned manner could be with more delays which results in much longer latency in the main memory access. In live migration, transferring memory information from source to destination without loss of data is very important. The less the data transmission repetition is, the less total live migration time, downtime and amount of transmitted data are.

Suitable Write caching and use a buffer during migration need a trickier policy. The best way for caching is disk writes into cache, and over time asynchronously flushes them to disk. Linux kernel first reads data from memory and then writes on cache then reads them from cache asynchronously. This has a significant effect on the speed of I/O, but there is a risk of data loss when data are not written on the disk. Therefore, in order to increase the performance, maximum effort should be done to preserve asynchronous communication.

5. PERFORMANCE EVALUATION

Results of the approach implementation are analyzed in terms of downtime, migration time and total transferred pages. During the experiments, transition VM is the only VM running on the source host and there are no VMs running on the target machine. The VM is transferred under different workloads. The VM is migrated between two physical hosts for at least three times [13]. The test environment used by the system configuration is presented in Figure 3.

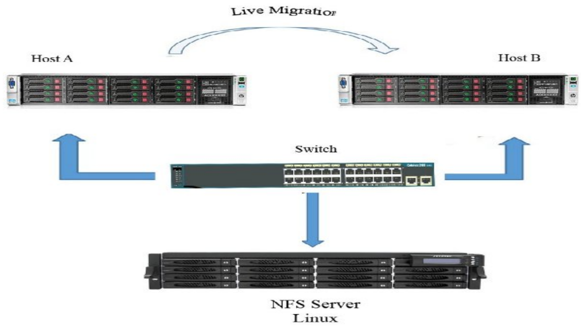

FIGURE 3. Experiment environment

Host and NFS Server with a 10 Gigabit Ethernet network are connected together. Ubuntu 12.04 with 3.0.23 kernel is used as the host OS [25, 26] and NFS Server and Win. XP SP3 for guest OS, as well as a VM monitor Xen 4.2.3. The VM migration is performed between two 580 G7 HP Servers. Each server has a 25 GB RAM and the Intel® Xeon 2.4GHz processor. Ubuntu with 4 GB RAM is used as NFS Server which shares guest's file system for other two physical hosts (Figure 3). All servers has 10 Gigabit Ethernet Network.

In addition, for PrmA-minthreshold and PrmB-minthreshold values, 5 and 70 percent are specified, respectively. These values are obtained after several experiments and led to the best performance. According to the experiment results, when VM workload is low, the downtime of the original migration approach of Xen [27] is about one second more than the proposed approach. When the VM is idle, such a difference is negligible. By increasing workload and memory dirtying rate, the proposed approach downtime is less than the Xen. Table 2 shows the comparison results of both the proposed and Xen approaches in terms of total migration time, downtime and total dirty pages.

TABLE 2.

Comparing the Xen and the proposed approaches

	Downtime (second)	Live migration time (second)	Total pages sent
Xen Approach	267	1234	33874
Proposed Approach	193	1227	17811

Figure 4 shows the live migration time during VM live migration in the Xen and proposed approaches. Using the Xen approach, without any specific workload and stress tool, live migration of VMs increased compared to the time of the proposed approach. This is due to the review pages and sort of dirty pages causing pressure on the structure and consequently increased migration time.

Figure 5 shows the number of transferred pages. When the VM run on the workload, and starts producing more dirty pages, the proposed approach takes maximum effort to communicate asynchronously and it is more organized and

ISSN: 2252-4274 (Print)
ISSN: 2252-5459 (Online)

regular compared to dirty pages which are copied from the source to the destination, thus the number of dirty pages is reduced to 47.4%.

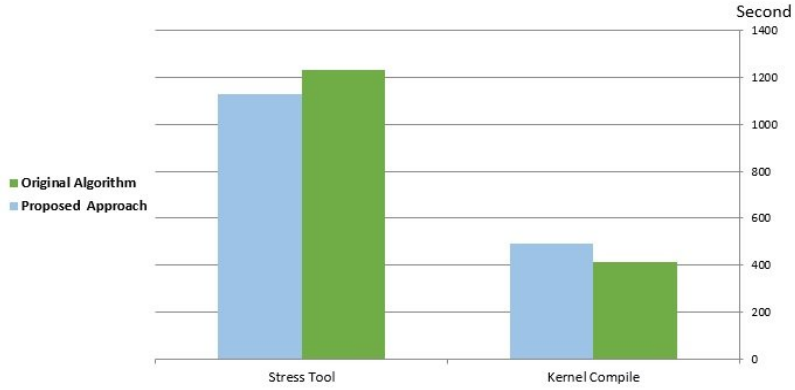

FIGURE 4. Comparing live migration time in the Xen and proposed approaches

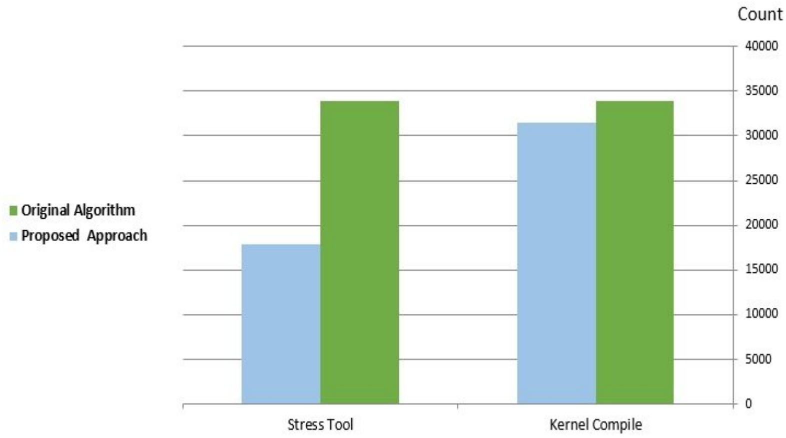

FIGURE 5. Comparing dirty pages produced in the Xen and the proposed approaches

By increasing workload and memory dirtying rate, the proposed approach downtime is less than the Xen approach. Figure 6 shows the VM downtime during live migration that is improved by 27.7% compared to the Xen approach.

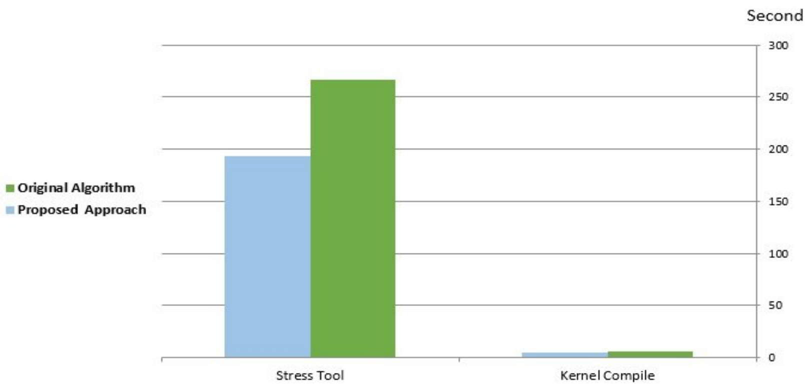

FIGURE 6. Comparing downtime in the Xen and proposed approaches

6. COMPARING THE PROPOSED APPROACH WITH SIMILAR APPROACHES

Currently, there is a lack of a standard benchmark suitable for live migration. In addition, the migration time in this paper depends on the pages in the page cache and IP priority. Thus, it is difficult to compare the evaluation results with the related works. Two approaches, namely “NRU and Modified Clock Policy” [22] and “LRU and Splay Tree” [23] have been proposed to prioritize dirty pages for live migration.

The LRU (Least Recently Used) and Splay Tree approach combines LRU cache with splay tree algorithm and performs in the pre-processing phase of the pre-copy method. As can be seen in Table 3, no improvement is reported for downtime in this approach and 27% improvement is reported for total transmitted pages.

TABLE 3.
Comparison between the proposed approach and the related works

Research Work	Live migration time	Downtime	Transferred page
Proposed approach	10	27/7	47/7
LRU and Splay Tree	11	Not reported	27
NRU and Modified Clock Policy	34	Without change	37

In NRU and modified clock policy approach, clock policy is used to investigate updated memory pages. In the pre-copy method, the less data transmission iteration implies both the better system performance and the less migration time. Therefore, all transmitted pages are checked continuously to make sure that updated pages are transmitted only once in each iteration process of the pre-copy method. As can be seen in Table 3, in NRU and modified clock policy method, downtime is not change and transmitted pages are improved by 37% while in the proposed approach, the down time and total transmitted pages are improved 27.7% and 47.7%, respectively.

7. CONCLUSION AND FUTURE WORK

In this paper, original migration approach of Xen was improved in terms of migration time, total transferred pages and downtime. When the VM is under heavy workload, the proposed approach has better performance compared to the Xen. According to the experimental results, by increasing the workload, the Xen approach sends more updated pages during live migration, but in the proposed approach, the number of sent pages might be reduced and consequently, migration time and downtime are improved.

In the future, it is possible to create an integrated priority management system for data centers. This system will handle the packets containing migration information and other traffics based on QoS requirements. In addition, implementing an approach for load balancing of packets with similar priority might improve the structure and QoS of the data.

ISSN: 2252-4274 (Print)
ISSN: 2252-5459 (Online)

REFERENCES

[1] S Akoush, A. Rice, R. Andrew, W. M. Andrew, H. Andy. Predicting the Performance of Virtual Machine Migration. In Proc. IEEE International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems, 2010.
[2] I Memon, L Chen, Q. A Arain, H Memon, G Chen. Pseudonym changing strategy with multiple mix zones for trajectory privacy protection in road networks. International journal of communication systems, October 2017
[3] I. Memon, Q. A. Arain, Dynamic path privacy protection framework for continuous query service over road networks, World Wide Web ,Volume 20, July 2017, Issue 4.
[4] I Memon, Q.Ali, A.Zubedi, F.A. Mangi. DPMM: dynamic pseudonym-based multiple mix-zones generation for mobile traveler. Multimedia Tools and Applications, Volume 76, November 2017, Issue 22.
[5] I. Memon, I. Hussain, R. Akhtar, G. Chen. Enhanced Privacy and Authentication: An Efficient and Secure Anonymous Communication for Location Based Service Using Asymmetric Cryptography Scheme. Wireless Personal Communications, Volume 84, May 2015, Issue 2.
[6] I. Memon, Q.A. Arain, H.Memon, F.A. Mangi. Efficient User Based Authentication Protocol for Location Based Services Discovery Over Road Networks. Wireless Personal Communications, Volume 95, August 2017, Issue 4.
[7] Q.i. Arain, D.ZhongLiang, I.Memon, A Zubedi, F.A. Mangi. Location Privacy with Dynamic Pseudonym-Based Multiple Mix-Zones Generation over Road Networks, Wireless Personal Communications, Volume 97, December 2017, Issue 3.
[8] Wenjin Hu, Andrew Hicks, Long Zhang, Eli M. Dow, Vinay Soni, Hao Jiang, Ronny Bull, Jeanna N. Matthews. A Quantitative Study of Virtual Machine Live Migration. In Proc. ACM Cloud and Autonomic Computing Conference, 2013, pp. 1-10.
[9] Gustafsson Erik. Optimizing Total Migration Time in Virtual Machine Live Migration. Thesis, 2013.
[10] S. Rawat, R.Tyagi, P. Kumar. An Investigative Study on Challenges of Live Migration. In Proc. IEEE Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO), Sept, 2016.
[11] A.Amani, K.Zamanifar. Improving the Time of Live Migration Virtual Machine by Optimized Algorithm Scheduler Credit. In Parallel, Distributed and Grid Computing, 2014, pp.346-351.
[12] S. Kumar Bose, S. Brock, R. Skeoch, N. Shaikh, S. Rao. Optimizing live migration of virtual machines across wide area networks using integrated replication and scheduling. In Proc. IEEE International Systems Conference, 2011, pp. 97-102.
[13] J. FarzinAlamdari, K. Zamanifar. A Reuse Distance Based Pre-copy Approach to Improve Live Migration of Virtual Machines. In Proc. 2nd IEEE International Conference on Parallel, Distributed and Grid Computing, 2012, pp.551-556.
[14] R. H. Michael, G. Kartik. Post-copy based live virtual machine migration using adaptive pre-paging and dynamic self-ballooning. In Proc. ACM

SIGPLAN/SIGOPS international conference on Virtual execution environments, 2009, pp.51-60.
[15] P. Getzi Jeba Leelipushpam, J. Sharmila. Live VM Migration techniques in cloud environment-A Survey. In Proc. IEEE Conference on Information and Communication Technologies, 2013.
[16] Nitishchandra Vyas, P.A.C. A Survey on Virtual Machine Migration Techniques in Cloud Computing. International Journal of Application or Innovation in Engineering & Management, 2016, p. 297-300.
[17] M. Shaikh, M.A. Memon, A. Shaikh, F. Deeba. Analyzing Virtual Machine Live Migration in Application Data Context. International Journal of Advanced Computer Science and Applications Volume 7, 2016, Issue 5.
[18] H. Liu, H. Jin, X. Liao, L. Hu, and C. Yu. Live migration of virtual machine based on full system trace and replay. In Proc. 18th International Symposium on High Performance Distributed Computing (HPDC'09), 2009, pp. 101– 110.
[19] P. Svärd, B. Hudzia, J. Tordsson. E. Elmroth. Evaluation of delta compression techniques for efficient live migration of large virtual machines. In Proc. the 7th ACM SIGPLAN/SIGOPS international conference on Virtual execution environments, 2011, pp. 111-120.
[20] K. Z. Ibrahim, S. Hofmeyr, C. Iancu, and E. Roman. Optimized pre-copy live migration for memory intensive applications. In Proc. International Conference for High Performance Computing, Networking, Storage and Analysis (SC), 2011, pp. 1-11.
[21] M. Fei, L. Feng, and L. Zhen. Live virtual machine migration based on improved pre-copy approach. In Proc. IEEE International Conference on Software Engineering & Service Sciences (ICSESS), 2010, pp. 230-233.
[22] Z. Safari, N. Bohlol and E. Fouladfar. Optimized live migration using NRU and modified clock policy. In Proc. 9th International Conference on e-Commerce in Developing Countries: With focus on e-Business (ECDC), 2015, pp. 1-8.
[23] EiPhyuZaw, Ni Lar Thein. Improved Live VM Migration using LRU and Splay Tree Algorithm. International Journal of Computer Science and Telecommunications [Volume 3, Issue 3, March, 2012].
[24] Cisco. dscp and precedence (2015). Available from: http://www.cisco.com
[25] The Linux kernel archive (2016). Available from: http:// www.kernel.org /
[26] Official Ubuntu Documentation (2015). Available from https://help.ubuntu.com
[27] How Does Xen Work (2015). Available from https://wiki.xenproject.org/

ISSN: 2252-4274 (Print)
ISSN: 2252-5459 (Online)