International Journal of Electrical and Computer Engineering (IJECE)
Vol. 9, No. 5, October 2019, pp. 4433~4440
ISSN: 2088-8708, DOI: 10.11591/ijece.v9i5.pp4433-4440

An overview of virtual machine live migration techniques

Artan Mazrekaj[1], Shkelzen Nuza[2], Mimoza Zatriqi[3], Vlera Alimehaj[3]
[1]Faculty of Contemporary Science and Technologies, South East European University, Republic of North Macedonia
[2]Institute of Natural and Applied Sciences, Dokuz Eylul University, Turkey
[3]Faculty of Electrical and Computer Engineering, University of Prishtina, Kosovo

Article Info

Article history:

Received Jul 18, 2018
Revised Apr 20, 2019
Accepted Apr 30, 2019

Keywords:

Cloud computing
Live migration
Post-Copy approach
Pre-Copy approach
Virtualization

ABSTRACT

In a cloud computing the live migration of virtual machines shows a process of moving a running virtual machine from source physical machine to the destination, considering the CPU, memory, network, and storage states. Various performance metrics are tackled such as, downtime, total migration time, performance degradation, and amount of migrated data, which are affected when a virtual machine is migrated. This paper presents an overview and understanding of virtual machine live migration techniques, of the different works in literature that consider this issue, which might impact the work of professionals and researchers to further explore the challenges and provide optimal solutions.

Corresponding Author:

Shkelzen Nuza,
Institute of Natural and Applied Sciences,
Dokuz Eylul University,
Boulevard: DEÜ Fen Bilimleri Enstitüsü Coğrafi Bilgi Sistemleri Anabilim Dalı Tınaztepe Kampüsü,
35160 Buca/İZMİR, Turkey.
Email: shkelzen.nuza@gmail.com

1. INTRODUCTION

Cloud computing is known an emerging paradigm where software, platform and infrastructure can be accessed as a service. In the cloud computing the key concept is Virtualization that allows sharing of a single instance of an application or a resource between multiple customers and organizations. Virtualization creates a virtual environment on a single physical machine by abstracting the hardware details. It allows us to use multiple instances of operating systems (known as guest OS) to handle the number of processes simultaneously and separately by each guest operating system [1]. It assigns a logical name correspond to a physical storage and provides a pointer to that physical resource when it is demanded. There are four known types of virtualizations:

1. Hardware virtualization: when the virtual machine (VM) software or the VM manager is installed directly on the hardware system.
2. Operating system virtualization: when the VM software or the VM manager is installed on the host OS and not directly on the hardware system.
3. Server virtualization: when the VM software or the VM manager is installed directly on the server system.
4. Storage Virtualization: grouping process of the physical storage from different multiple network storage devices to look like a single storage device.

As virtualization splits a physical machine (PM) into several VMs, this brings us to the term of the VM which is a software implementation of an environment of computing where a program or OS can be installed and run [2]. VMware ESX / ESXi [3], Virtual PC [4], Xen [5], and Microsoft Hyper-V [6], KVM [7], VirtualBox [8] known as hypervisors which are some popular virtualization software. Xen and VMware

hypervisors have special technology for live migration, and they are known as XenMotion and VMotion. Feng et al. [9] compare the performances of both and the given results when migrating identical VMs show that VMotion generates less data transferred than XenMotion. Live migration of VMs offers the possibility for allocation of resources to running services without interruption during migration process that is important for services with particular Quality of Service (QoS) requirements [10].

Background

Live migration of VMs is a process of migrating the states (CPU, memory, storage, network, etc.) of VM from one physical machine to another one. On the performance of live migration, various techniques make various impacts. In general, live migration of VMs has several benefits including:

1. *Online maintenance*, sometimes to enhance reliability and availability of a system, it must be connected to clients so all VMs are migrating away without being disconnected.
2. *Load Balancing*, when the load is considerably unbalanced, the VMs should be migrated from PMs that are overloaded to other PMs that are not in the overloaded state.
3. *Manageability and maintenance*, movements of VMs and shutdown of PMs for maintenance.
4. *Energy Management*, consolidation of VMs, switch off underutilized PMs to reduce data center's heat loss and power consumption.
5. *Improved performance and reliability*, the application performance will not be degraded.
6. *Minimum violation of Service Level Agreement (SLA), meeting the SLA requirements between cloud providers and cloud users.*

There are many papers that handles the VM migration techniques, but this paper is intended to make a better understanding of VM migration techniques.

In [11], on different hypervisors are compared live migration efficiencies. Xen spent the most downtime while KVM spent the least sum of downtime for storage and memory live migrations.

In [12], is represented *pre-copy* approach that shows better performance compared with pure *stop-and-copy*. The *pre-copy* combines many rounds of *push* and short *stop-and-copy* at the end. The *stop-and-copy* has high downtime. The *pre-copy* technique is better than *on-demand*, because *stop-and-copy* has high total migration time, and computer resources in straight proportion with increasing the time. The most dominant parameter on performance is the migration link speed.

Also, in [13] are summarized the advantages of different approaches, such as *pre-copy* and *stop-and-copy*. The performance of live migration can be affected differently using trade-off techniques on different hypervisors. Using different virtualization techniques on different hypervisors, authors has compared, CPU usages, memory utilizations and transfer time. In [14], the authors show that applying *pre-paging* technique improves the page fault problem. Working sets that will be used in the future are predicted, and pages are loaded before being accessed, then normal *post-copy* live migration is processed. In order to eliminate the migration of free memory pages, a mechanism *dynamic self-ballooning* will be executed. The results from pre-copy on Xen are compared with the improved *post-copy* approach and it shows that *post-copy* is better in connection with number of transferred pages and total migration time, while *pre-copy* is better in terms of downtime.

In order to make the optimal selection for destination VMs, in [15] is used a cost model on hypervisor Xen. For the calculation, the different parameters are used, such is the sum of network traffic, size of VM's memory and degree of dirty pages. During the creation of a cost-aware migration algorithm [16] use a Shanon's notion of entropy [16]. By calculating the tradeoff of performance influence and migration time the algorithm wants to minimize migration cost.

In [17], Live Gang Migration explains, where the group of VMs migrates simultaneously by QEMU/KVM hypervisors with a minimum total network traffic and migration time. In order to do concurrent live migration of co-located VMs is used De-duplication technique. De-duplication deals with elimination of redundant information. In order to detect similarity of contents page identical detector uses *hash* function. The result of this paper shows that Live Gang Migration can reduce total network traffic and migration time. In order to reduce the migration load by reducing set of the data to migrate, suggests new model named Incremental. It uses *block-bitmap* in order to synchronize all write accesses to the local disk [18].

Problem Description

There have been considerable contributions in the live migration of VM techniques. For the techniques of VM live migration that have been presented so far, in order to find an optimal solution of dynamic consolidation in the cloud environments, many approximation techniques have been used. By comparing the major techniques of VM live migration in the cloud environments, it has enabled us to understand the impact factors and limitations that emerge from these techniques, thus making the narrower definition of the issues related to dynamic consolidation and resource utilization, such as a very complex

issue in cloud infrastructure. The main focus of this paper is to provide to the researchers comprehensive understanding and classification of key aspects of live migration techniques, on resources like CPU, memory, network, and storage.

Contributions

In this paper, the problem of dynamic consolidation of VMs through the live migration mechanism is considered, which enables efficient resource allocation in the cloud environments. The main parameters that have a direct impact on the efficiency of the resources that are tackled, such as downtime, disruption time, total migration time, amount of migrated data, and performance degradation. Furthermore, in terms of live migration, the major techniques that have an impact on resource utilization are dealt with rigorously as memory, file, network, and device migration.

This paper is organized as follows. Section 2 explains the performance metrics. In the section 3 are tackled the live VM migration techniques, whereas in the section 4 are given types of other VM live migration techniques. The paper concludes with a summary and a comparison of related work.

2. PERFORMANCE METRICS

The following metrics are mostly used to measure the efficiency and performance of a VM live migration process.

2.1. Downtime

This metric represents the interval of time while services are not running and available. Actually, is the time duration from when VM pauses on the source PM till it resumes on the destination PM.

2.2. Disruption time

It is the time when clients that are connected to the services that are running on the migrated VM notice degradation of service responsiveness. Therefore, when clients request any service, response time takes longer. This means, disruption time is the time period during which services on the Virtual Machine show lower performance to the client because of the migration process. Also the methods for synchronization and the transfer rates have an influence on this performance metric.

2.3. Total migration time

This metric represents the time duration since the migration starts till the states/services on the source and destination PM are totally synchronized. To reduce total migration time it is preferred for decreasing the size of transferred data, for example to compress the transferred data before we send it.

2.4. Amount of migrated data

It is the amount of data that are transmitted during the whole time of migration. Sometimes it can be referred to as pages transferred. The minimal amount is considered the size of the run time states, like storage size, memory size, CPU state size, etc. In most cases it will be greater than the actual run time state size. This will not happen in freeze-and-copy method because in this method must be some redundancy for protocols and synchronization.

2.5. Performance degradation

This metric represents the decrease of performance of the service which is caused during the migration process. Performance degradation is evaluated by comparing the throughput of service during the migration and without it.

3. LIVE VM MIGRATION TECHNIQUES

3.1. Memory migration

Memory migration is a process of migrating VM memory instance from the source to the destination. The process of memory migration can be divided into phases [19]:

1. Push phase: the hypervisor transfers memory pages to the destination PM when VM on source is still running. Dirtied pages in transmission process are sent again till the rate of those recopied pages are more than dirtying rate for consistency.
2. Stop-and-copy phase: after being copied the memory pages by source VM who has been stopped than start a new VM to the destination. This means the source VM has stopped until pages has been copied to the destination VM, and then the new VM has started.

3. Pull phase: After execution of the VM on destination PM starts if a page required is not found, then a page fault occurs, so the page is pulled from the source VM through the network.

All the migration techniques try to reduce total migration time and down time. There are two main approaches in memory migrations: a) Pre-Copy and b) Post-Copy.

3.1.1. Pre-copy approach

In the pre-copy approach all memory pages are sent before the VM is resumed to execute on the new node. This technique has been implemented on many kinds of hypervisors like Xen, VMware and KVM and uses iterative *push* and *stop-and-copy phase* [20]. Due to the iterative procedure of *push phase* some memory pages, which are known as dirty pages will be modified. Then these pages are regenerated on source PM during iterations of the migration process. In the second phase, termination depends on the defined threshold. The termination phase is executed if any one of conditions meet [21]:

a. Number of iterations exceeds the number of iterations pre-defined,
b. The total amount of memory which has been sent, or
c. The number of total dirty pages in previous round falls below the defined threshold.

In the pull phase, at source PM the VM migrating process is suspended, and after that remaining dirty pages and processor's state are suspended. When the migration process is completed correctly, hypervisor resumes VM migration on the destination PM.

Guangyong et.al. [22], proposed two techniques for improving pre-copy approach: 1) technique of memory compaction based on disk cache and memory snapshot; 2) scheme of adaptive downtime control based on the history of VM's memory update information called Writable Working Set - WWS. They have implemented the method in KVM hypervisor. It shows that the memory compaction techniques can reduce memory transfer time by a factor of 2 and mostly in the first phase. Experimental results proved that the adaptive VM downtime control technique successfully handled the live migration for the VM running memory intensive workloads. But when the memory contents of target migrating VM change too much, the memory compaction technique might not work well.

3.1.2. Post-copy approach

In this approach [23], each memory page has been transferred once and this makes it better than the pre-copy approach. The total migration time and number of transferred pages are less [24]. At the beginning, *post-copy* suspends the migrating VM at the source PM, it copies minimal processor state to the destination PM, resumes the VM, and starts fetching memory pages from the source PM through the network. The way how pages are fetched, helps rising different variants of *post-copy*, where each is considered as a function for improvements.

a. Demand paging: when starts the migrated VM on destination PM, and the pages it needs are not in the memory, then page fault occurs. It can be serviced by requesting the page through the network. The page is transferred and due to the traffic Virtual Machine slows down.
b. *Active paging*: while VM is running, the pages are proactively pushed into it [25]. If any page faults, those can be serviced by demand paging and for transferred pages no page faults occur.
c. *Pre-paging*: this variant is like *active paging* but predicting the special locality of VM memory access pattern [23]. The next pages which will be accessed are transferred to the VM.
d. *Dynamic Self-Ballooning (DSB)*: is used to avoid transfer of free memory pages. It releases periodically free pages of VM back to the hypervisors, so this way speeds-up the process of migration with negligible performance degradation. Thus, releasing process of these pages that are not used is increased to the destination PM, as a result of this the total migration time is reduced.

3.1.3. Hybrid technique (Pre and post copy)

It uses the benefits of pre-copy and post-copy approaches, and the combination of these two approaches reduces service downtime and the total migration time. First, it works as a pre-copy approach while VM is running on the source PM. After the first iteration of VM memory transfer is stopped, and it resumes at destination PM with its processor state and dirty pages. Then, the remaining pages are transferred by post-copy approach [24].

Choudhary *et.al.* [21], compares *pre-copy* with *post-copy*, seen that the second approach reduces the total migration time and the transferred number of pages. But due to migration latency of fetching the pages it has more downtime than the first approach. Another disadvantage of *post-copy* is that if any failure occurs during the migration, the recovery may not be possible. When pre-copy or post-copy improves the performance, depends upon the workload type and performance goal of migration. It is concluded that pre-copy could be considered as a better approach for read intensive workload, and post-copy for write intensive or large memory workload.

Hines *et.al.* [23], also compared post-copy and pre-copy approach on Xen. They show improvements in some migration metrics like pages transferred, network overhead and total migration time using a range of VM workloads. They mitigate using of post-copy with adaptive pre-paging to eliminate duplicate of all page transmissions. Yingwei *et.al.* [26], describes a scheme for whole-system live migration. For achieving an inconsiderable downtime and finite dependency from source PM, this kind of migration transfers the whole system run-time state, like memory data, local disk storage and CPU state of the VM. It proposes a TPM – Three - Phase Migration algorithm, which composed of *pre-copy*, *freeze-and-copy* and *post-copy*.

3.2. File migration

A consistent view and a location independent view of file system should be available on all PMs to support VM migration. A solution to support this, is providing each VM with its own virtual disk, which is mapped in the file system, and it transports this virtual disk's contents along with the other states of the VM. Another proposed way could be to have a global file system through all PMs, where could be located a VM, but it is not so practical to provide a consistent global root file system through all PMs.

In [27] is tackled a distributed storage technology for VM migration, which is known as Internet Suspend/Resume (ISR). It lets suspend on one PM and seamlessly resume on another PM. A distributed file system used in ISR serves to transfer the files of suspended VM state. Hypervisor uses local files to store the contents of each VM's virtual disk and then transfers those and other information of VM state to the destination. There are known two different techniques, one called *smart copying* and other one is a *proactive state transfer*, which helps to reduce the amount of data that have to be transferred from suspended VM to resume VM.

Also, another system which tackles the file migration is *Zap* [26] that provides a thin virtualization layer that introduces a *process domain* abstraction known as *pod* and is set on top of the OS. It provides a set of processes with the same virtualized view of the system and is indicated by a virtualized file system private and corresponding private file system namespace that presents this group. When *pods* are created or they are moved to a PM, also is created a *pod* identifier on PM for each *pod* to serve as an area for virtual file system of *pod*. *Zap* takes care to make the directory inaccessible for processes on the PM which are not in given pod.

3.3. Network migration

For remote communication of systems with VM, each VM should have its MAC address and the virtual IP address. Hypervisors perform mapping of virtual IP and the MAC addresses to their corresponding VMs. Also, is provided an unsolicited ARP reply from migrating PM which advertises the IP moved to a new location if the machines included in VM migration are connected with switched network. So, future packets are sent to the new location by reconfiguring all the peers. The migrating operating system has an actual MAC address to detect its move to a new port [28]. There is a system for network migration, which is known as Quasar [29], provides support for migration of computing environments. It is equipped with a virtual network mechanism which makes migration transparent to guest OSes. The mechanism allows network connections to be kept across migration.

3.4. Device migration

Nowadays, virtualization is not supported in the most of the hardware, so device virtualization could only rely on pure software technology [30]. Physical resources are shared using software based virtualization between different guests, by preventing access to device resources from guests. Device migration needs that VMs can't use PM specific devices as some devices are difficult to migrate. Generally, there are three types of device support provided to make device migration possible [28]:

a. *Emulation*, it serves for emulating a device in software. For example, a virtual console could be registered as /dev/console.
b. *Virtualization*, if a VM migrates to another PM which has an equivalent device, the VM will be able to utilize it.
c. *A Non-migratable device driver*, passes all requests to the device on the PM, but when the device is in use it does not allow migration.

4. OTHER LIVE VM MIGRATION TECHNIQUES

4.1. Adaptive memory compression

For balancing the cost and the performance of VM migration, authors at [31] introduce an adaptive zero-aware compression algorithm. They proposed the VM migration technique called MECOM that uses

memory compression in order to fast, and stable VM migration. From experiments that compared to Xen, their system can reduce 32% of total migration time, 27.1% of downtime and 68.8% of total transferred data.

4.2. Using shared storage

In order to keep the downtime to minimum the authors at [32] presents a technique which will reduce the total time to migrate a running VM from one PM to another. Their technique makes an updated mapping of memory pages that presently exist in the same form on the storage device and tracks the VM's I/O operation to the network attached storage device. Through the iterative pre-copy phase, the memory to disk mapping will be sent to the destination PM, after that it will get the contents from the network attached storage device, rather than transferring pages from the source to the destination. From the results it is seen that a reduction of up over 30% in total transfer time for a range of benchmarks.

4.3. Exploiting data de-duplication

Zhang et.al [33] describes a VM migration approach, Migration with Data De-duplication (MDD), which presents data de-duplication into the migration. MDD has hash based fingerprints to discover identical memory pages, and in order to exclude redundant memory data when migration and utilizes the self-similarity of run time memory image uses Run Length Encode (RLE). From the experiments, it is seen that MDD has reduced 34.93% of total migration time, 56.60% of total data transferred during migration, and 26.16% of downtime.

4.4. Energy aware virtual machine migration

The paper [34], introduces an algorithm that deal with the load of the PM and which efficiently migrates the VMs. There are other important factors that are used to select VMs which will be migrated, so the energy consumption will be minimized by shutting down underutilized PMs. This will cause the reduction of the energy cost. The authors evaluate their proposed solution while using their own simulator. After the experiments their results show that proposed method reduces energy consumption up to 20.8 % for static VM load and up to 22.0 % of dynamic VM load compared to pure performance based VM migration.

4.5. Continual migration

Cui et.al [35] represent that by continuously propagating state of the VM's to a backup PM through live migration techniques, applications in the VM with minimal downtime can be repaired from hardware failures. From the results it is seen than in a continual migration system, if a failure is detected, the VM can be repaired in less than 1 sec, although performance impact to the protected VM can be reduced to 30%.

4.6. Asynchronous replication and state synchronization

Liu et.al [36] explains that execution trace is logged on the source PM, in order to coordinate the running source and destination VMs to achieve consistent state, where is used a synchronization technique. The authors introduce the implementation of a new approach, CR/TR-Motion, which adopts check-pointing/recovery and trace/replay technologies to provide fast, transparent VM live migration for WAN and LAN. The consumption of the network bandwidth and migration downtime can be reduced by CR/TR-Motion. Experiments show that the approach can reduce migration compared to memory-to-memory technique in a LAN, up to 31.5% on total migration time, up to 72.4% on application noticed downtime, and up to 95.9% on the data to synchronize the VM state. For a variety of workloads migrated across WANs, the application performance overhead for migration is 8.54 percent on average.

4.7. Gang migration

Live migration of multiple VMs at the same time from one group of PMs to another in reaction to events such as unavoidable failures load spikes is known as Gang Migration (GM). A large amount of network traffic generated by Gang migration causes overload on core network links and switches in a datacenter. Using global de-duplication, the authors in [37] present a technique to keep down the network overhead of GM. The GM determines and excludes the retransmission of duplicate memory pages between VMs running on numerous PMs in the cluster. The implementation of GM global de-duplication and evaluate it using QEMU/KVM VMs. From the results it is seen that it reduces the total migration time of VMs and the network traffic. As well, has a smaller reverse performance impact on network-bound applications.

In the Table 1 is given a summary of several techniques for VM Live migration. In this summary are considered the most important features on live migration, such as: basic migration technique, name of the technique, hypervisor used, metrics, and achievements/benefits.

Table 1. Comparison of VM live migration techniques

Basic migration technique	Name of the technique	Hypervisor used	Metrics	Achievements/Benefits	Ref
Pre-copy Algorithm	Shared storage	Xen	- Migration time - Downtime - Network traffic	32% total migration time; Number of pages transferred is almost equal; 37% of downtime.	[19, 24, 32]
	Adaptive Memory Compression	Xen	- Number of pages transferred	32% total migration time; 68.8% number of pages transferred reduced; 27.1% downtime.	[31]
Post-copy Algorithm	Adaptive pre-paging	Xen	- Number of pages transferred	Eliminate all duplicate page transmissions; Reduce the number of network-bound page faults to within 21%.	[23, 25, 14] [33]
Pre-copy algorithm	Exploiting Data De duplication	Xen	- Migration time - Downtime	Reduces 56.60% of total data transferred during migration; 34.93% of total migration time.	
Live migration Pre-copy	Energy aware VM algorithm	Custom built simulator simulator	Power consumption	Reduces Power Consumption, 28% on static load and 22% on dynamic load.	[34]
Live migration Pre-copy	Continual migration	KVM	Fault tolerance	Hardware failure (recovered less than one sec); Reduced Downtime, 30% overall performance.	[35]
Live migration pre-copy	Asynchronous replication and state synchronization	UMLinux/ Re-Virt-log and replay tool	Downtime Migration time Network traffic fault Tolerance	Reduces 72,4% downtime; Reduces 31,4% migration time; 95.9% network bandwidth.	[36]
Live migration	Gang migration	QEMU/KVM	Migration time Network bandwidth	Reduces 42% migration time; Reduces 65% network bandwidth	

5. CONCLUSION

This paper is an overview of live migration of VM techniques. The live migration involves the process of moving VM or multiple VMs from one physical machine to another, while they're running. Services that are running on VM's must be available to the users, so they will be migrated while they are running. The reasons for live VM migration are: system maintenance, load balancing, power management, proactive fault tolerance, resource sharing.

The paper focuses on the comprehensive literature review of other work refers and trying to bring to the researchers' understanding of live migration techniques by description of weaknesses and strengths, key aspects of migration like CPU, memory, network, and storage. The discussion is concentrated in some of performance metrics, like: downtime, total migration time, performance degradation, etc., that affect the process of VM live migration. Classification of live migration techniques by explaining three basic ones: pre-copy, post-copy and hybrid live VM migration.

REFERENCES

[1] G. Singh and P. Gupta, "A review on migration techniques and challenges in live virtual machine migration," *2016 5th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)*, Sep 2016, pp. 542-546.
[2] T. Erl, R. Puttini, and Z. Mahmood, *Cloud Computing: Concepts, Technology & Architecture*, P. Hall Press, 2013.
[3] "VMWare, "vSphere ESX and ESXi Info Center," [Online], Available: https://www.vmware.com, [accessed on May 2018].
[4] Microsoft, "Windows Virtual PC," [Online], Available: http://www.microsoft.com/windows/virtual-pc, [accessed on June 2018].
[5] Xen, "Xen Hypervisor," [Online], Available: http://www.xen.org/products/xenhyp.html, [accessed on May 2018].
[6] Microsoft, "Hyper-V Server 2012," [Online], Available: www.microsoft.com/server-cloud/hyper-v-server, [accessed on June 2018].
[7] KVM, Kernel-based Virtual Machine," [Online], Available: www.linux-kvm.org, [accessed on June 2018].
[8] Oracle, "VirtualBox," [Online], Available: www.virtualbox.org, [accessed on May 2018].
[9] X. Feng, J. Tang, X. Luo, and Y. Jin, "A performance study of live VM migration technologies: VMotion vs XenMotion. Network Architectures, Management, and Applications IX," *Asia Communications and Photonics Conference and Exhibition (ACP)*, Nov 2011.
[10] Q. Zhang, E. Gurses, R. Boutaba, J. Xiao, "Dynamic Resource Allocation for Spot Markets in Clouds," *The 11th USENIX conf. on Hot topics in management of internet, cloud, and enterprise networks and services*, ACM, 2011.

[11] W. Hu, A. Hicks, L. Zhang, E. M. Dow, V. Soni, H. Jiang, R. Bull, and J. N. Matthews, "A Quantitative Study of Virtual Machine Live Migration," *In ACM Cloud and Autonomic Computing Conference*, New York, Aug 2013.
[12] S. Akoush, R. Sohan, A. Rice, A. W. Moore, and A. Hopper, "Predicting the Performance of Virtual Machine Migration," *IEEE International Symposium on Modeling, Analysis Simulation of Computer and Telecommunication Systems (MASCOTS)*, pp. 37–46, 2010.
[13] G. Soni and M. Kalra, "Comparative Study of Live Virtual Machine Migration Techniques in Cloud," *Int. J. Comput. Appl*., vol. 84, pp. 19-25, Dec 2013.
[14] M. R. Hines, U. Deshpande, and K. Gopalan, "Post-copy Live Migration of Virtual Machines," *ACM SIGOPS Oper Syst Rev*, vol. 43, pp. 14-26, Jul 2009.
[15] M. Forsman and A. Glad, "Automated Live Migration of Virtual Machines," *Msc thesis, Blekinge Institute of Technology*, Karlskrona, Sweden, Sep 2013.
[16] X. Qin, W. Zhang, W. Wang, J. Wei, X. Zhao, and T. Huang, "Towards a Cost-Aware Data Migration Approach for Key-Value Stores," *IEEE International Conference on Cluster Computing*, Oct 2012, pp. 551–556.
[17] U. Deshpande, X. Wang and K. Gopalan, "Live Gang Migration of Virtual Machines," *in Proceedings of the 20th International Symposium on High Performance Distributed*, New York, pp. 135-146, Jun 2011.
[18] Y. Luo, B. Zhang, X. Wang, Z. Wang, Y. Sun, H. Chen, "Live and incremental whole-system migration of virtual machines using block-bitmap," *IEEE International Conference on Cluster Computing*, Sep 2008, pp. 99–106,.
[19] C. Clark, K. Fraser, S. Hand, J. G. Hansen, E. Jul, C. Limpach, I. Pratt, and A. Warfield, "Live Migration of Virtual Machines," *Proceedings of the 2nd Conference on Symposium on Networked Systems Design & Implementation*, vol. 2, pp. 273–286, May 2005.
[20] D. Kapil, E. S. Pilli, and R. C. Joshi, "Live Virtual Machine Migration Techniques: Survey and Research Challenges*," 3rd IEEE International Advance Computing Conference (IACC)*, 2012, pp. 963-969.
[21] A. Choudhary, M. Chandra Govil, G. Singh, L. K. Awasthi, E. S. Pilli, and D. Kapil, "A critical survey of live virtual machine migration techniques," *Journal of Cloud Computing: Advances, Systems and Applications*, vol.6 (3), 2017.
[22] G. Piao, Y. Oh, B. Sung, and C. Park, "Efficient Pre-Copy Live Migration with Memory Compaction and Adaptive VM Downtime Control," *IEEE Fourth International Conference on Big Data and Cloud Computing*, Dec 2014.
[23] R. M. Hines and K. Gopalan, "Post-Copy Based Live Virtual Machine Migration Using Adaptive Pre-Paging and Dynamic Self-Ballooning*," Proceedings of the 2009 ACM SIGPLAN/SIGOPS international conference on Virtual execution environments*, Mar 2009, pp. 51-60.
[24] S. Sharma, and M. Chawla, "A technical review for efficient virtual machine migration," *International Conference on Cloud and Ubiquitous Computing and Emerging Technologies (CUBE)*, Sep 2013.
[25] D. Perez-Botero, "A Brief Tutorial on Live Virtual Machine Migration from a Security Perspective," University of Princeton, USA, 2011.
[26] S. Osman, D. Subhraveti, G. Su, and J. Nieh, "The Design and Implementation of Zap: A System for Migrating Computing Environments," *Proceedings of the 5th Symposium on Operating Systems Design and Implementation*, Boston, MA, Dec 2002.
[27] M. Satyanarayanan, B. Gilbert, M. Toups, N. Tolia, A. Surie, D. R. O'Hallaron, A. Wolbach, J. Harkes, A. Perrig, and D. J. Farber, "Pervasive Personal Computing in an Internet Suspend/Resume System*," IEEE Internet Computing*, vol. 11 (2), 2007.
[28] S. Venkatesha, S. Sadhu, and S. Kintali, "Survey of Virtual Machine Migration Techniques," *University of California, Santa Barbara*, Mar 2009.
[29] K. Onoue, Y. Oyama, A. Yonezawa, "A Virtual Machine Migration System Based on a CPU Emulator," *Proceedings of the 2nd International Workshop on Virtualization Technology in Distributed Computing*, Nov 2006.
[30] E. Zhai, G. D. Cummings, and Y. Dong, "Dong Live Migration with Pass-through Device for Linux VM 2008," Linux *Symp.*, vol. 2, pp. 261-267, 2008.
[31] H. Jin, L. Deng, S. Wu, X. Shi, X. Pan, "Live virtual machine migration with adaptive memory compression," *IEEE International Conference on Cluster Computing and Workshops*, Oct 2009.
[32] C. Jo, E. Gustafsson, J. Son, B. Egger, "Efficient live migration of virtual machines using shared storage," *Proceedings of the 9th ACM SIGPLAN/SIGOPS international conference on Virtual execution environments*, vol. 48 (7), pp. 41-50, Jul 2013.
[33] X. Zhang, Zh. Huo, J. Ma, and D. Meng, "Exploiting Data Deduplication to Accelerate Live Virtual Machine Migration," *IEEE International Conference on Cluster Computing*, Oct 2010.
[34] M. H. Al Shayeji and M. D. Samrajesh, "An Energy-aware Virtual Machine Migration Algorithm," *Second International Conference on Advances in Computing and Communications (ICACC2012),* Aug 2012.
[35] W. Cui, D. Ma, T. Wo, and Q. Li, "Enhancing reliability for virtual machines via continual migration," *15th International Conference on Parallel and Distributed System*, Dec 2009, pp. 937-942.
[36] H. Liu, H. Jin, X. Liao, C. Yu, and C.Z. Xu, "Live virtual machine migration via asynchronous replication and state synchronization," IEEE *Transactions on Parallel and Distributed Systems*, vol. 22 (12), 2011.
[37] U. Deshpande, B. Schlinker, E. Adler, and K. Gopalan, "Gang Migration of Virtual Machines using Cluster-wide Deduplication," *13th IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing*, May 2013.