THOMSON REUTERS ENDNOTE

[Kiran., 3(5): May, 2016] ISSN 2349-4506
Impact Factor: 2.265

Global Journal of Engineering Science and Research Management

CLOUD MULTI-TIER APPLICATION MIGRATION MANAGEMENT SYSTEM

Kiran N*, Kumar H.R
*M.Tech Department of Computer Science Shridevi Institute of Engineering and Technology Tumakuru, Karnataka, India
Assistant Professor Department of Computer Science Shridevi Institute of Engineering and Technology Tumakuru, Karnataka, India

DOI: 10.5281/zenodo.51759

KEYWORDS: Cloud, live migration, multi-tier application, virtual machine.

ABSTRACT

Virtualization Technology (VT), re-invented to address most of the computer systems resource utilization challenges especially for Cloud Environment. An important feature of VT is live migration of the Virtual Machine (VM) that consists of Guest Operating Systems and applications running on it. Enabled by virtualization technologies, various multi-tier applications (such as web applications) are hosted by virtual machines (VMs) in cloud data centers. Live migration of multi-tier applications across geographically distributed data centers is important for load management, power saving, routine server maintenance and quality-of-service. Different from a single-VM migration, VMs in a multi-tier application are closely correlated, which results in a correlated VM migrations problem. Current live migration algorithms for single-VM cause significant application performance degradation because intermediate data exchange between different VMs suffers relatively low bandwidth and high latency across distributed data centers. In this paper, we design and implement a coordination system for correlated VM migrations in the cloud. Particularly, we propose an adaptive network bandwidth allocation algorithm to minimize the migration cost in terms of migration completion time, network traffic and migration downtime. Experiments using a public benchmark show that coordination system significantly reduces the performance degradation and migration cost of multi-tier applications.

INTRODUCTION

INTERNET applications have been prosperous in the era of cloud computing, which are usually hosted in virtual machines in geographically distributed data centers. Live migration of Internet applications across data centers is important for different scenarios including load management, power saving, routine server maintenance and quality- of-service. Additionally, Internet applications tend to have dynamically varying workloads that contain long-term variations such as time-of-day effects in different regions. It is desirable to move the interactive/ web application to the data center that has better network performance to users for lower response time [4]. Also, workloads can be migrated across different data centers to exploit time-varying electricity pricing [2]. The recent advance of VM live migration techniques is able to relocate a single VM across data centers with acceptable migration cost. Typical Internet applications employ a multi-tier architecture, with each tier providing certain functionality. Specific to multi-tier applications, we need to migrate several tightly-coupled VMs in multi-tiers, instead of a single VM. Previous studies have demonstrated the potential performance penalty of multi-tier applications during migration [1]. In this paper, we investigate whether and how we can reduce the migration cost without suffering application performance degradation.

A typical multi-tier web application consists of three tiers: presentation layer (web tier), business logic layer (App tier) and data access layer (DB tier). Different layers usually run on different VMs and have different memory access patterns. VMs are correlated because only when all VMs of the multi-tiers are migrated to another data center, they can completely and efficiently serve requests in that data center. We call this problem correlated VM migrations. Correlated VM migrations can cause significant performance penalty to multi-tier applications. Consider the following scenario: if the middle tier is first migrated, then the other two tiers must redirect the communication and data access traffic to another data center and wait for the processing results to be sent back. Moreover, because the multi-tier application and migration processes share the same link for data transferring, given the data-intensive nature of multi-tier applications and limited network bandwidth between two data centers, network bandwidth contention may cause significant performance degradation for both applications and VMmigrations. While live migration of VMs provides the ability to relocate running VMs from one physical host to another without perceivable service downtime, the state-of-theart VM migration techniques mainly target a single VM (either within a data center or across different cloud data centers). These techniques cannot fundamentally solve the correlated VM migrations problem. We need effective and efficient mechanisms to coordinate correlated VM migrations across distributed data centers.

The principal characteristic feature of virtualization enabled architecture is its dual state hardware- Privileged and Non-privileged access. The former makes all the instructions available to the user, whereas the latter has to make supervisory calls to the operating system nucleus, in order to have privileged access. The main characteristics of a VM is that it runs and uses only the resources allocated to it and does not go beyond that. Virtualization of the instances of operating systems are highly useful as this feature facilitates the datacenter managers to provide isolated software and hardware environments to balance the user loads in a secure, reliable and fault tolerant way. Multiple VMs own the portion of the underlying hardware resources and each VM run their own separate operating systems which are handled and managed by the Virtual Machine Monitor (VMM). Many privileged and critical instructions are executed by these VMMs on behalf of the VMs running on it

The need to migrate the VMs arises because of thee overloading of one physical machine that runs multiple VMs onto it. Consequently, there is overheating which may degrade the performance and may also lead to faulty operations and system crashes. Hence a conditional load balancing is required for better manageability of the cluster of servers [2]. There are few advantages of migration of a whole of VM. The narrow interface between a virtualized OS and the virtual machine monitor (VMM) avoids the problem of residual dependencies. There are various ways to migrate a VM from one physical node to another. “Pure stop-and-copy” or “Cold” migration technique halts the VM and copies all its associated memory pages to another pre-identified destination node and then resumes the VM on it. On the contrary, few selected hypervisors like Xen and VMWare do a “Live” or “Hot” migration. The advantage of the latter method is that even the concerned applications and processes are unaware of the VM migration

BACKGROUND

The cloud services providers while focusing on amazing user experience of their services also stress on the need of optimized usage of their resources and data durability of their users. So they developed various algorithms and implementation to migrate a VM in various cases like excessive CPU requirement, memory constraints, Stagnant/idle, etc.

There are few techniques to resolve such issues which include VM migration and Process Migration. On Xen hypervisor, the VM migration feature can easily be run and effects analysed with xm-migrate command. And this command has been highly used and emphasized on since it can be modified with variety of attributes that goes with the command such as whether we want the migration to be live or cold. But the real problem arises on detecting when to fire the VM migration mechanism and detecting which VM is the causing the trouble.

Process migration demonstrates a functionality of transferring a process running on one machine to the other. But, there is an inherent difference in the operating concepts of virtual machine migration and process migration [4]. Though in practice, migrating the process is difficult and quite complex as it should take care of legacy applications and at the same time it should also leverage the currently installed and related large databases of operating systems and maintain independence on different machines. These can be overcome by using a VMM based migration. VMMs such as VMware use Hardware abstraction to encapsulate the complete OS environment in such way that it can be suspended from one machine and resumed onto the other one provided there are inherent similarities in the system architectures of the operating systems. But VM migration supersedes Process Migration except in some cases that occur due to the narrow interface between a virtualized OS and the virtual machine monitor (VMM) where it avoids the problem of residual dependencies. VM migration has the advantage of transferring internal memory states in a very consistent and efficient way.

Another part is that the System Virtual Machines (VMs) are widely used from personnel computers to large organizations. System virtualization acts as powerful means of abstraction for upcoming applications. On the cloud computing platform resources are provided according to need on the principle of pay per use. To accommodate specific requirements of subscribers and how the balance is maintained between Cost, Quality and Resources is mentioned in the Service Level Agreement (SLA). The Cloud Service Providers guarantee a level and quality of service to the users as per the terms and conditions of the SLAs. A lot of challenges are faced while catering the need of the users and at the same time making efficient use of underlying heterogeneous resources in a dynamic and efficient way, which is inherently expected from Cloud Services in terms of Infrastructures, platforms and software.

_A._ Virtual Machine Migration

Virtualization is a technology for cloud computing environments. It allows partitioning one physical machine into multiple virtual machines that runs concurrently and also shares the same physical resources. Virtual Machines (VM) can be provisioned on-demand, replicated and migrated. It has the ability to migrating virtual machine from one physical machine to another machine for the purpose of load balancing and also to make the physical machine fault tolerant. Virtual Machine consolidations can be used to reduce power consumption of cloud data centers. For instance, through server consolidation, multiple (virtual) servers can be allowed to run simultaneously on a single physical server [6].

B. Live Virtual Migration.

Live migration is emerging as an impressive technology to efficiently balance load and optimize VM deployment across physical nodes in a data center. With the help of live migration, VMs on a physical node can be transferred to another without shutting down the system.

This is useful in various scenarios.

- VMs on a failing physical node can be migrated to a healthy one.
- Idle VMs on a node can be migrated to others for optimizing resource utilization.
- VMs on a stressed physical node can be migrated across various nodes for load balancing.

On a broad level, live migration process can be classified into two steps - (i) switching the control to the destination (ii) transferring the data (memory/disk) to the destination. The two most common methods of live migration can be easily differentiated based on different iterations of the steps mentioned above.

- Pre-copy - Memory is transferred first and then the execution is transferred.
- Post-copy - Execution is transferred first and then the memory.

1) Pre-copy

The approach behind pre-copy is to transfer the memory to the destination over a series of iterations. The newly written pages are transferred in each iteration and this process is repeated until either the limit on iteration reaches or the final data is too small for causing any network transfer overhead.

1) Memory and VCPUs are reserved on the destination host.
2) When the migration is issued, a check on page writes is initiated and all the RAM contents are transferred to the destination. This is the first iteration.
3) In the subsequent steps, only the pages that have been dirtied since the last iteration are transferred until the iteration limit is reached or the memory of dirty pages in an iteration is low enough.
4) The execution of VM on source is stopped and CPU state, registers, Virtual devices state and last memory pages are transferred to the destination.
5) VM is resumed at the destination. Pre-copy is well proven for read intensive workloads but in case of write intensive VMs, large set of dirty pages result in performance degradation. In worst cases, pre-copy won't even work. This is possible when the data write rate is more than the network bandwidth.

2) Post-copy

In contrary to pre-copy, post copy transfers the VCPU and devices state on the destination in the first step and starts the execution on the destination in the second.

C. Static load balancing

Static Load Balancing (SLB) refers to the load balancing algorithm that distributes the load strictly on the basis of certain predefined rules relating to the nature of input loads. It does not consider which node is receiving more or less load. In all static algorithms final selection of the virtual machine is done immediately after creation of application. Further it cannot be changed while in execution. These static load balancing techniques are suitable for a system in which load is limited and request of the clients is also limited. But nowadays load on cloud servers is also increasing and also the load is not static hence we need more efficient algorithms then static load balancing algorithms.

CORRELATED VM MIGRATION PROBLEM

This section introduces the background of single-VM migration techniques and correlated VM migrations.

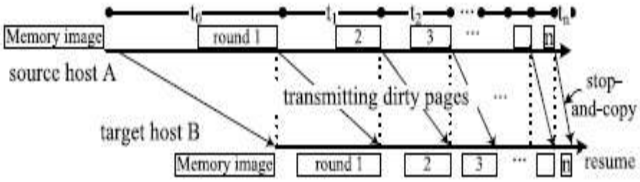

Fig.1.Live migration algorithm performs pre-copying in iterative rounds.

Single-VM migration. Live migration of VMs has been an effective approach to manage workloads in a non-disruptive manner. Among different implementations of VM live migration, pre-copying algorithm is the most popular and widely used in today's virtualization platforms. Thus, this paper focuses on pre-copying algorithm in Xen, and the methodology in our paper can be extended to other VMmigration techniques.

The basic workflow of pre-copying algorithm in Xen is described as follows. As shown in Fig. 1, memory precopying is conducted in several iterative rounds. The VM's physical memory is first transferred from host A to host B, while the source VM continues running in host A. Pages dirtied in each round must be iteratively re-sent in the next round to ensure memory consistency. That is, the data to be transmitted in each round are dirty pages generated in the previous round. After several rounds of pre-copying, a stop-and-copy phase is performed to transmit the remaining dirty pages while the source VM temporarily stops execution. When the final data transferring is done, the VM on host B resumes and takes over the VM on host A.

Correlated VM migrations. The objective of this paper is to solve the correlated VM migrations problem raised in multi-tier applications. As illustrated in Fig. 2, a multi-tier web application is migrated from DC 1 to DC 2. Each tier is running on multiple VMs, and thus the VMs across different tiers are correlated with data dependency. Network bandwidth is a critical resource across distributed data centers. It is usually much smaller than the network bandwidth within a data center. Previous studies assumed that the network bandwidth between two data center was 465 Mbps. Without loss of generality, we assume that the peak network bandwidth between DC 1 and DC 2 reserved for the migration processes is B. As discussed in Introduction, the application traffic and migration traffic share the same links between two data centers. The bandwidth contention between them may result in significant performance degradation in both application and VMmigration.

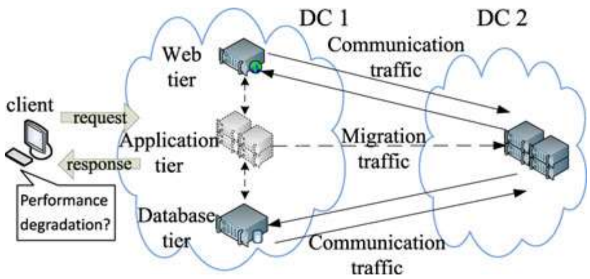

Fig. 2. Performance penalty due to live migration of a multi-tier web application across distributed data centers.

ISSN 2349-4506
Impact Factor: 2.265
Global Journal of Engineering Science and Research Management

SYSTEM IMPLEMENTATIONS

Steps evolved in migration Live migration means the migration of a VM from source physical machine to destination when the virtual machine is powered up. Virtual machine migration should occurs in such a way that it should minimize both total migration time and down time. Downtime is the time for which service is not available. Downtime is transparent to the users. Total migration time is the time from when the migration is started to when the source virtual machine is discarded. Source physical machine is taken for maintenance, repair and upgrade [3]. The tradeoffs between the requirements of total migration time and downtime become easy by dividing memory placement into following phases:

Push phase: In this phase, the VM on source keeps running while required phases are placed to the destination through the network. To maintain consistency, the pages which are change during migration must be sent again.

Stop and Copy phase: In this phase the source virtual machine is stopped. The pages are transferred from source to destination and the new virtual machine is started. The time gap between halting the virtual machine on source host and initiating the VM on the destination host is called downtime. Downtime depends upon the memory size and applications executing on the virtual machines. It is from few milliseconds to seconds. Some techniques like PDF (probability density function) exist to minimize downtime.

Pull phase: In pull phase, the new virtual machine is created and if it need a page which is not present, then the page fault occurs. And the page is pulled from source across network. The virtual machine migration is done by suspending the virtual machine at source. When the virtual machine is suspended at the source, required state of CPU, memory and registers is transferred at the destination. The virtual machine is resumed at destination, the whole memory state is not moved yet, small amount of data still exist on the source. When the needed page is not present on the destination host, then page faults are occurred, known as network fault. Source machine recover network fault by sending the required pages across the network. Pure demand paging and pre paging is used. Live virtual machine is highly used in virtualized data centers and IT departments. It separate physical servers and software from each other and offers features like dynamic load balancing and online server maintenance etc.
Following logical steps evolved in migration:

Stage 0 Pre-migration: Virtual machine migration is initiated with an active virtual machine on source host A. To speed up future migration process, a destination host is pre-selected and the required resources for migrating VM should be guaranteed.

Stage 1 Reservation: The request is generated to move an operating system from physical host A to physical host B. First of all it is confirmed that the required resources are present on host B and virtual machine container of same size is reserved. If the required resources are not there then the VM continuously run on host A.

Stage 2 Iterative Pre copy: During first iteration, the whole memory pages are moved from host A to host B. Next iteration only copy those pages which becomes dirty during previous phases.

Stage 3 Stop and Copy: In this phase running operating system instance is suspended at host A and the network traffic is redirected from host A to host B. The remaining memory pages and CPU state are transferred. At the end of stop and copy stage, both host A and host B has consistent suspended copy. The pages at a are considered primary and are resumed if any type of failure occurs.

Stage 4 Commitment: Indication is given by host B to host A that the consistent operating system image is received. Acknowledgement is given by host A and original virtual machine is discarded by host A. Now host B is primary host.

Stage 5 Activation: The migrated virtual machine on host B is activated. Device drivers are reattached to the destination machine and post migration code advertise IP address.

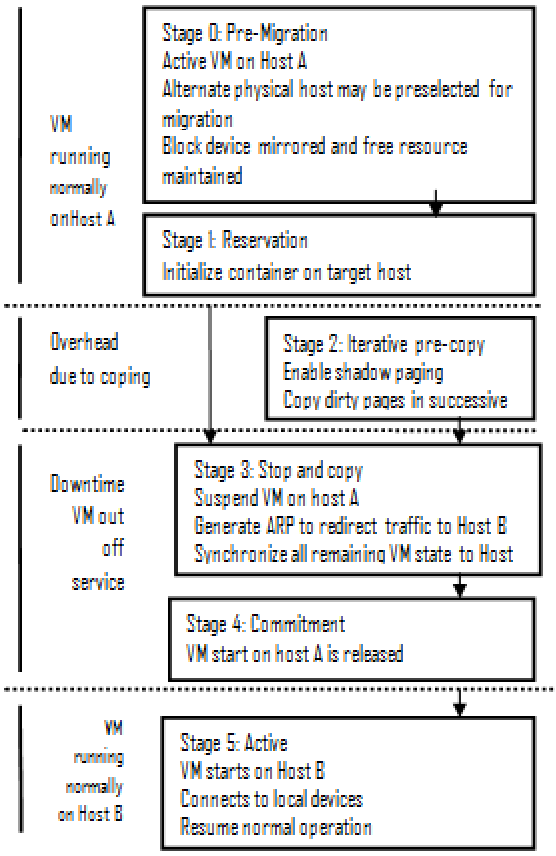

Fig. 3. Migration Timeline

CONCLUSION

The migration of a VM is a tricky and complex process. It may lead to loss of VM or corruption of the same if not handled properly. This innovative technique of trigger based VM migration will surely provide users to take more control over its VM and help them to better manage their VMs, consequently, a better and efficient way of managing the underlying system resources, hence bolstering the core idea of virtualization, which is the technical backbone of cloud computing.

A dynamic load balancing strategy and algorithm is developed and implemented on Xen VMM for VM load balancing based on Network File System (NFS). According to the current states of VMs, it computes and identifies in advance, the most suitable VM where the upcoming application can be allocated. The overhead involved for identifying suitable VMs for successive allocations is drastically reduced as the previous and current states are already available. Additionally, this strategy is quite efficient and requires less computational overhead as compared to normally used Migration strategies and other traditionally adopted load balancing techniques and hence better resource utilization comparatively.

REFERENCES

1. Rimal, B.P., Choi, E., Lumb, I., 2009, "A taxonomy and Survey of Cloud Computing Systems" 25th International Joint Conference on INC, IMS and IDC.
2. Secant Method, (2003) [Online]. Available: http://math.fullerton.edu/mathews/n2003/SecantMethodMod.html

3. Cloud Workload Patterns, (2010) [Online]. Available: http://watdenkt.veenhof.nu/2010/07/13/workload-patterns-for-cloudcomputing
4. S. Al-Kiswany, D. Subhraveti, P. Sarkar, and M. Ripeanu, “VMFlock: Virtual machine co-migration for the cloud,” in Proc. ACM Symp. High Perform. Parallel Distrib. Comput., Jun. 2011, pp. 159–170.
5. N. Bila, E. Lara, K. Joshi, H. A. Lagar-Cavilla, M. Hiltunen, and M. Satyanarayanan, “Jettison: Efficient idle desktop consolidation with partial VM migration,” in Proc. ACM Eur. Conf. Comput. Syst., Apr. 2012, pp. 211–224.
6. 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,” in Proc. ACM Symp. Operating Syst. Principles, Oct. 2003, pp. 164–177.
7. J. Billaud and A. Gulati, “hClock: Hierarchical QoS for packet scheduling in a hypervisor,” in Proc. ACM Eur. Conf. Comput. Syst., Apr. 2013, pp. 309–322.
8. Mishra, M.; Das, A; Kulkarni, P.; Sahoo, A, 2012, "Dynamic resource management using virtual machine migrations", Communications Magazine, IEEE, Vol. 50, No.9, pp.34-40.
9. Agarwal, A. and Raina, S. 2012,”Live Migration of Virtual Machines in Cloud”, International Journal of Scientific and Research Publication, Vol. 2, No. 6, pp 1-5.
10. Christopher, C., Gorm, H.J., Steven, H., 2005,”Live virtual machines migration,” 2nd conference, Networked Systems Design Implementation: USENIX.
11. Jiuxing, L., Panda, D. K., Wei, H., Qi., G., 2007, ”High performance virtual machine migration,” in IEEE Conference.
12. Yingwei, L., Binbin, Z., Xiaolin, W., Zhenlin, W., Yifeng, S., and. Haogang, C., 2008, ”Live and incremental whole-system migration of virtual machines using block-bitmap,” in IEEE International Conference on Cluster Computing, pp. 99-106.