Improving Virtual Machine Migration in Federated Cloud Environments

Antonio Celesti, Francesco Tusa, Massimo Villari and Antonio Puliafito
Dept. of Mathematics, Faculty of Engineering, University of Messina
Contrada di Dio, S. Agata, 98166 Messina, Italy.
e-mail: {acelesti,ftusa,mvillari,apuliafito}@unime.it

***Abstract*—Cloud federation is the future of the cloud computing. In such new emerging scenarios, the "hot" and "cold" migrations imply the movement of a virtual machine disk-image from a server placed in a cloud provider to a server placed in another one, with a consequent consumption of bandwidth and cloud resources. This paper aims to reduce such costs proposing a Composed Image Cloning (CIC) methodology. Our approach does not consider the disk-image of a VM as a single monolithic block, but as a combination between "composable" and "user data" blocks. Since the former may be cloned locally in the destination clouds, the amount of data which have to be transferred may be reduced. In order to test our strategy we implemented a real test bed extending the OpenQRM platform. Experiments show the validity of the proposed approach in large-scale federated cloud environments, where the amount of data transferred significantly reduces when the number of live migrations increases.**

***Keywords*-Cloud Computing; Federation; Virtual Machine disk-image relocation; Virtual Machine Cloning; OpenQRM.**

I. Introduction

Cloud Computing [1] relies its computational capabilities on the concept of "virtualization". Virtualization technologies aim to hide the underlying infrastructure by introducing a logical layer between the physical infrastructure itself and the computational processes. Through Virtualization Machine Monitors (VMMs commonly known as "hypervisors"), i.e. processes that run on top of a given physical host, each cloud is able to control and emulate several processing environments named virtual machines (VMs), each running its own "guest" software, typically an operating system. Commonly a Cloud, aggregating one or more VMs, provides its clients three types of on-demand services: Infrasturcture as a Service (IaaS), Platform as a Service (PaaS), and/or Software as a Service (SaaS). Each VM, is constituted of a VM disk-image, in a given hypervisor format (e.g. Xen [2], KVM [3], VMware [4]), basically containing the file system and the guest operating system.

Virtualization allows a cloud to relocate a VM from a physical host to another inside its infrastructure, exploiting the VM migration concept. In a federated scenario where clouds cooperate together to obtain new business opportunities (cost-effective assets optimization, power saving, and on-demand service allocation), VMs are not enclosed within the cloud boundaries, but can migrate from a cloud provider to another. VM migration, from a source cloud to a destination one over a Wide Area Network (WAN) implies to transfer three elements: memory, status, and storage (the disk-image containing file system and user data). This may represent a bottleneck as it implies to transfer GB and GB of data over the network, with a consequent waste of resources, both in the source and the destination Clouds.

In this paper, we focus on the dynamic VM allocation, proposing a "Composed Image Cloning" (CIC) methodology able to reduce the cost of the VM disk-image relocation over a WAN. The paper is organized as follows. Section II provides an overview of the most adopted VM migration techniques in both LAN and WAN environments. Section III highlights the problems deriving from both the dynamic VM allocation and VM migration in federated cloud environments and introduces our methodology. In Section IV we demonstrate how our approach is able to reduce the cost of VM disk-image relocation. Our experimental results are discussed in Section V. Conclusion and highlights to the future are presented in Section VI.

II. Related Works and Background

VM migration consists of transferring a VM from a source to a destination physical host. Basically there are two kinds of VM migration: "hot" (or live) and "cold" migration. In "hot" migration the VM does not lose its status and the user does not perceive any change. In "cold" migrations the VM loses its status and the user notices a service interruption. The downtime is referred to the time elapsing from the instant when the VM is turned off in the source host and the moment in which the same VM is turned on in the destination host. If the downtime is negligible and the VM status is maintained, the migration is defined "hot", otherwise "cold". Considering the live migration, several works are available in the literature. In [5], the authors present a pre-copy technique where the VM's memory content is the first element to be migrated followed by the processor states. An alternative "win-win" post-copy strategy for the VM migration across a Gigabit LAN is presented in [6], where the authors defer the transfer of the VM's memory content until after its processor state has been sent to the target host. Since, in such works, the live migration occurs inside the same cluster, both techniques do not migrate the VM disk-image, which is typically hosted inside a shared Network Attached Storage (NAS). The cost

of the VM live migration in a cloud environment is discussed in [7], where the authors evaluate the effect of live migration on the performance of applications running inside Xen based VMs.

Cloud federation implies the dynamic load management of VMs. In [8] a management algorithm is proposed that allows to decide about the reallocation of VMs in cloud contexts characterized by a large number of hosts. Such algorithm simply identifies some critical instances and takes decisions without recurring to typical thresholds.

Regarding "hot" or "cold" migration over WAN, due to the network latency, it is not possible to utilize a shared NAS for the VM disk-images, which consequently has to be relocated too. Some studies exist that have addressed the VM disk-image migration. Similarly to [9], some of them assume a pre-arrangement of the VM disk-image before the VM migration occurs, but in our opinion, such solution is not suitable to highly dynamic federated cloud scenarios, where changes occur quickly. Instead, a storage access mechanism capable of rapidly relocating VM disk-images, with a limited impact on I/O performance of the migrant VMs, is presented in [10] and [11]. The VM disk-image relocation occurs by means of a xNBD target server placed in the source cloud and a xNBD proxy server placed inside the destination cloud. When a VM is migrated, its disk-image is relocated from the source to the destination cloud NAS. The VM disk-image is divided into many pieces: first are transferred the pieces with high-priority (on which the migrated VM needs to perform instantaneous I/O operations), and in background the ones with low-priority. When a VM needs to perform an I/O operation, the request is intercepted by the xNBD proxy server of the destination cloud which copies the required pieces of disk-image from the xNBD target server of the source cloud into the destination cloud NAS. The process continues in background until the whole VM disk-image is transferred.

In our opinion the best practice for the WAN live migration is combining the pre-copy or post-copy technique [5] with the storage access mechanism for relocable VM disk-images [10].

III. Our Composed Image Cloning Approach

Considering WAN environments, in this paper, unlike many works already available in literature, we do not try to reduce the downtime due to a live migration (as in [5], [7]), nor to propose a new storage access mechanism able to rapidly relocates VM disk-images between source and destination clouds (as in [10], [11]), but we specifically focus on how to reduce the size of a VM disk-image which has to be transferred for both "hot" and "cold" migrations.

Considering federated scenarios with thousands of cloud providers interconnected over the Internet, we think the VM disk-image movement, due to either a "hot" or "cold" VM migration, represents a practice that, over the time, could become too expensive: it implies the copy of GB and GB of data over the network with a waste of resources in both the source and the destination clouds. In the following, we specifically focus on such problem presenting our "Composed Image Cloning (CIC)" methodology able to reduce the size of the VM disk-image, and consequently the amount of data which have to be transferred from a source to a destination federated cloud due to a VM migration.

A VM disk-image is an image of a VM storage according to a given hypervisor format including a kernel, an OS, programs, and user data. Commonly, when a client requires to allocate a VM on a cloud provider, two strategies are available: I) the client decides to upload an existing VM disk-image in the cloud virtualization infrastructure; II) the client chooses to create a new VM in the cloud virtualization infrastructure selecting a set of predetermined features (i.e. CPU, Memory, OS, programs) by means of the cloud's user interface. In the first case, the uploaded VM disk-image can include also the user data, whereas in the second case, the cloud provider, according to the client's preferences, chooses from its repository the VM disk-image template fitting the preselected client requirements and allocates the target VM. Once the new VM is allocated and is up, the client begins to customize it with his personal user data. In both cases, the VM disk-image can be considered as a "monolithic block". When a VM migration has to be performed from a source to a destination federated cloud, such a "monolithic block" has to be entirely transferred.

Considering the aforementioned case, when a client needs to allocate a new VM machine on a cloud provider, the target VM is usually allocated, by the cloud, cloning a master VM disk-image contained in its repository. Cloud providers use different hypervisor formats and various methods to store and upload these master VM disk-images from other repositories. However there are some drawbacks associated to the storage of master VM disk-images in each cloud repository. When a cloud receives a VM allocation request, it checks in its repository an appropriate template. If the cloud does not find it, the cloud needs to obtain a new master VM disk-image template. For example, if a client requires a VM with kernel 2.6.32, Ubuntu, and mysql server, but the cloud holds a template T1 (kernel 2.6.32, Suse, and mysql server), since each VM disk-image template is considered as a monolithic block, the request can not be satisfied. In order to satisfy further requests for VMs with such features, the cloud has to "prepare" a new monolithic template T2 (kernel 2.6.32, Ubuntu, and mysql server). Though T1 and T2 hold common features (kernel 2.6.32, mysql server), since they are independent and monolithic blocks, there is a waste of storage resources in the cloud repository.

According to our analysis, considering a VM, we can identify both "modular" and "non-modular" parts. Modular parts are well-know elements, which compose a VM disk-image (i.e. hypervisor format, kernel, OS, and programs);

non-modular parts (optional) are the user data. Our CIC solution allows to save storage resources in the cloud repository and reduce the amount of data transferred due to VM disk-image relocation over a WAN. CIC is based on three main concepts: Modularity, Composability, and Self-Learning. CIC is founded on the assumption that a VM may be seen as a set of modular parts (Modularity). The cloud repository does not contain a set of VM disk-images anymore, but a set of VM Components (VM-CMPs). We define a Composed Virtual Machine (CVM) as a VM machine made of a set of VM-CMPs (Composability). When a cloud needs to allocate a new VM, it looks for an appropriate VM disk-image template in its repository. If it does not find a template which matches the features of the required VM, it looks for a new appropriate template. This occurs also if the cloud holds other similar templates, causing a waste of storage capabilities. With our approach, since the cloud manages VM-CMPs, rather than the entire VM disk-image, the cloud has only to obtain the required VM-CMPs for a CVM allocation. Such practice is analogous to the usage of shared libraries already adopted by computer programs. Shared libraries are particular libraries, placed inside the OS, that are loaded by programs when they start. When the shared libraries are installed properly, if needed, all programs that start afterwards can automatically use them. If the required libraries are missing, programs can obtain them from an external repository. Considering our CIC approach, once a VM-CMP is obtained from other external repositories it can be used for subsequent CVM allocations. Therefore, a cloud, gaining experience, learns how to compose new types of CVM (Self-Learning).

IV. Advantages in VM Disk-Image Relocation

In this Section, we show how to improve the VM disk-image migration in federated cloud environments, using our CIC methodology.

A. Composable and User Data Blocks

Assuming a VM live migration in a WAN environment such as a cloud federation, once the migration of both status and memory has occurred, it is also needed to transfer the disk-image. Commonly, as depicted in Figure 1, a VM migration implies the relocation of the entire disk-image which can be considered as a monolithic block.

Instead, as depicted in Figure 2 we consider the VM disk-image as composed of two blocks: the "composable block" and the "user data block". The former is a CVM disk-image, whereas the latter can be seen as a work directory where the user data of the VM owner are stored. When the VM is up, this latter is attached to the root file system (rootfs) and becomes accessible from the OS.

Therefore, our methodology assumes each VM is composed of two elements: the CVM disk-image and the user work directory, both stored inside a NAS of the cloud where

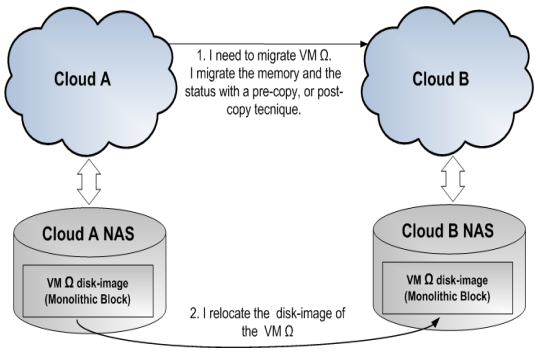


Figure 1. VM disk-image relocation in WAN environments.

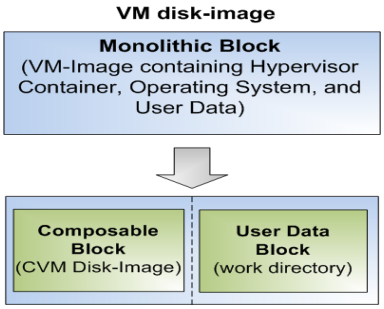


Figure 2. VM disk-image seen as a set of *Composable* and *User Data* Blocks.

the VM is hosted. In case of VM migration in a LAN environment, our model does not add any improvement compared to the existing migration techniques because the migration affects only memory and status (the disk-image is not moved). Instead, considering the VM migration in a federated cloud scenario, spread in a wide geographical area interconnected over the Internet, our model permits to reduce the clouds' resource usage and the bandwidth consumption in terms of VM disk-image relocation. Figure 3 depicts an example of VM live migration between federated cloud providers A and B.

Each cloud holds its own VM-CMP repository and NAS. The repository contains a set of VM-CMPs, whereas the NAS contains the storage of each VM (the composable and user data blocks). As depicted in Figure 3, cloud A VM-CMP repository contains A, C, D, E, F, G, and Z "VM-CMPs", whereas cloud B repository contains A, C, D, H "VM-CMPs". Cloud A, for some reason (for example power saving) needs to migrate the VM Ω (formed by A, Z, C, D "VM-CMPs") into cloud B. Cloud B checks in its own repository and notifies Cloud A it does not hold the VM-CMP Z. Therefore, cloud B copies from the repository of cloud A the VM-CMP Z, and once it holds all the required VM-CMPs, it locally clones the CVM disk-image of the VM Ω. Thus, cloud A can migrate the memory and status with some technique (pre-copy [5] or post-copy [6]), and then the user data block with some storage access mechanism

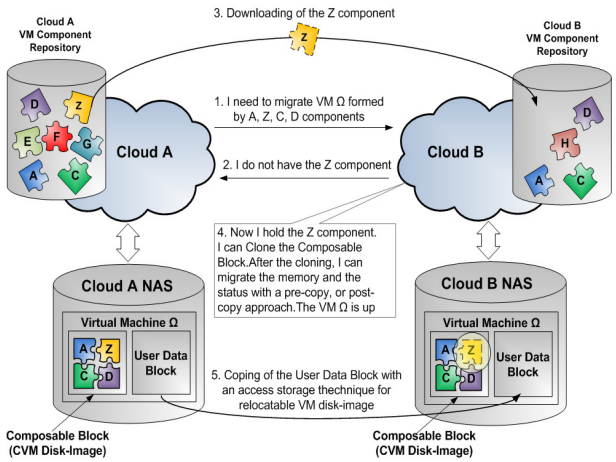


Figure 3. Example of VM storage migration using The CIC methodology.

[10]. Therefore, using our approach, the relocation of a VM disk-image from a cloud to another does not consist in the movement of a monolithic block anymore, but, considering the best case (further details are provided in the Section IV-B), only in the movement of the user data block.

B. The Best, Average, and Worst Cases

When a source cloud needs to migrate a VM into a destination cloud, three different situations may occour: *the worst case*, *the average case*, and *the best case*. In order to better clarify the ideas, we consider two clouds: source and destination. The source cloud wants to migrate three VMs: VM1, VM2, VM3. For each VM, we consider its respective composable block as a set of VM-CMPs: $CVM_{VM1} = \{W, X, Y, Z\}$, $CVM_{VM2} = \{A, B, C, D\}$, and $CVM_{VM3} = \{A, B, G, F\}$. Moreover, we define the set $C_R = \{A, B, C, D, H, I, L\}$ as the VM-CMPs repository of the destination cloud.

The "worst case" is when the destination cloud does not have within its repository any VM-CMP for the local arrangement of the composable block (CVM disk-image), therefore, it has to copy all the required VM-CMPs from the source cloud repository. In such case the amount of storage transferred and the workload is the same of the storage access mechanism [10], adopted for the VM disk-image relocation over a WAN: the destination cloud has to copy all the VM-CMPs in its repository, plus the user data block. An example is when the source cloud migrates VM1. Since $CVM_{VM1} - C_R = \{W, X, Y, Z\}$, the destination cloud has to copy the W, X, Y, Z "VM-CMPs" from the source cloud repository.

The "best case" is when the destination cloud has in its repository all the required VM-CMPs, and it has only to copy the user data block. An example is when the source cloud migrates VM2. Since $CVM_{VM2} - C_R = \{\emptyset\}$, the destination cloud does not have to copy any VM-CMP from the source cloud.

The "average case" deserves further considerations. It is when the destination cloud has in its repository only some of the required VM-CMPs for the locally composition of a given composable block and therefore it needs to obtain all the missing pieces. An example is when the source cloud migrates VM3. Since $CVM_{VM3} - C_R = \{G, F\}$, the destination cloud has to copy only the G and F VM-CMPs from the source cloud. Considering a scalable federated cloud scenario, when a cloud joins the federation, initially, the average case is closer to the worst case because the cloud does not have a varied and stocked set of VM-CMPs in its repository. Instead with the experience gained over the time and by means of the "Self-Learning" due to the previous live migrations of VM from other source federated clouds, the VM-CMP repository of the destination cloud grows up becoming always more varied and stocked. Therefore, for large scale scenarios, through the gained experience of each cloud, the average case becomes closer to the best case.

V. Experiments and Performance Analysis

In this Section, after a brief description of the testbed employed for testing our new CIC migration approach, we present and analyze a set of numerical results collected during the execution of a set of experiments. A comparison between our migration approach and a simple VM disk-image transfer is reported.

A. OpenQRM Improvements for the VM Storage Migration Using our CIC Methodology

Keeping in mind the idea of our Composed Image Cloning (CIC) solution, and considering the existing projects acting as distributed computing middleware, we identified OpenQRM (Open Qlusters Resource Manager) [12] as the one whose features better match our idea of CVMs: initially created for commercial purposes, it is an Open Source Data Center Management Platform for automation and management of scalable clusters.

In order to accomplish such task, OpenQRM is grounded on the concept of *appliances*, representing the services which should be provided by the datacenter. Regarding our idea of CVM, an appliance can be considered as the set of "composable blocks" constituting the modular part of the VM storage (i.e. the CVM-Image) previously stated. According to the OpenQRM terminology, an appliance practically is the combination of different components: a *kernel* (a Linux Operating System kernel), an *image* (i.e. a root-file system of a supported Operating System), a *resource* (the hypervisor container or even a physical resource) and *service requirements* plus *service-level-agreements* (SLA). An innovative feature of OpenQRM is that it supports VM migration between different hypervisor formats.

Also in the OpenQRM approach, as well as previously stated in our CVM model, user data is not directly included

within the appliances. It represents, in fact, the VM storage part we already named *User Data Block* which is located on a separated storage system within the cluster (e.g. a NAS).

Although the idea on which OpenQRM relies was created for Datacenter management, it can be employed in a WAN federated cloud environment as well, in order to minimize the amount of data transferred during a VM migration. In our opinion, together with its powerful capabilities related to Datacenter management, OpenQRM is also able to address the service migration within a federated Cloud, gaining the advantages of our CIC methodology previously exposed. When a migration has to be performed in a WAN environment, instead of transferring the whole VM disk-image, it is more convenient to clone a new VM on the destination cloud according to the CIC approach.

Obviously, since the original version of OpenQRM does not include any feature regarding cloud federation, a set of integrations has been performed on the project, in order to allow the communication of different (remote) clusters (datacenters): a new specific module, named *Migration Agent*, has been added to the OpenQRM architecture to allow inter-cluster communication over the WAN. As depicted in Figure 4, according to this new model, when a VM disk-image relocation has to be performed from a remote OpenQRM cloud to another, their respective Migration Agents establish a connection through XMPP [13], performing all the operations needed to pursue the WAN VM disk-image migration. Since the destination cloud holds all the VM-CMPs needed to perform the CVM-image cloning, only the User Data Block has to be transferred.

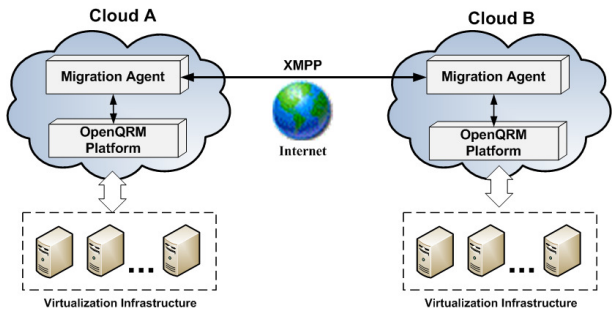


Figure 4. OpenQRM Integrations

XMPP allows to gain knowledge about all the available clouds of the federation and retrieve other sites' availability information, merely analyzing the specific *XMPP room* where the clouds aiming to take part into the federation hold a subscription. When the cross-site migration has to be performed, the following sequence of steps will be executed:

1) On the OpenQRM source cloud, a specific virtual machine (CVM) is selected for the migration (the Virtual Machine is also marked with a UUID in order to be univocally identified in the Federated Cloud;
2) The migration agent on the OpenQRM source cloud, employing the XMPP presence concept, choses the OpenQRM destination cloud;
3) Once the destination cloud has been found, a migration request containing the CVM features will be forwarded from the source site to the destination cloud;
4) If the previous step is properly accomplished and the destination cloud is able to satisfy the request, the CVM is "cloned" according to the specified requirements (i.e. kernel version, rootfs, image container, etc.)
5) The User Data Block attached to the Virtual Machine on the source location, has to be transferred and mounted on the destination cloud.

During our experimental trials, we arranged a testbed installing OpenQRM on two different clusters; in the second phase, we conducted a series of experiments on the testbed itself iterating the above mentioned sequence of steps and collecting the related measures, as reported in Section V-C.

B. Testbed Specifications

In order to evaluate the validity of our proposed solution, we setup two different (federated) clouds: the first one (cloud A), located at the University of Messina, is composed of a set of computers whose characteristics are Dual-Core AMD Opteron Processor 2218 HE with 8 GB of RAM; the second one (cloud B), located in the same metropolitan area (but belonging to a different administration - Inquadro srl) is constituted of a set of computers with similar hardware characteristics.

On each cloud, the cluster is composed of several physical hosts on which the Linux Operating System was installed: Debian 5.0 lenny was installed in the cloud A hosts, whereas Ubuntu 8.10 Intrepid Ibex was installed on the cloud B hosts. Both clusters include, for each host, the following set of software modules: 1) openQRM Server 4.5 and plugins (with package openqrm-server-entire); 2) VMware Server 2; 3) Kernel-based Virtual Machine (KVM), already included within the kernel; 4) Logical volume Manager (LVM); 5) Network system Server (NFS); 6) PHP/Java Bridge 5.5.4.

Cloud A has been connected to the Internet by means of the GARR network having 135 Mbps of bandwith, whereas cloud B has been connected to the Internet through an ADSL connection with 120 Kbps of bandwidth.

C. Results

In this Section, we compare two different approaches of performing the VM migration. During our experiment, we first considered the migration of a monolithic VM disk-image (containing both the OS and user data), carried out transferring data from the source cloud A to the destination cloud B, and measuring the time needed to accomplish the whole process. In a second phase, we performed a VM migration employing our CIC approach, also measuring the involved timings. Both the VM disk-images contain a

similar set of software modules and are able to run the same services.

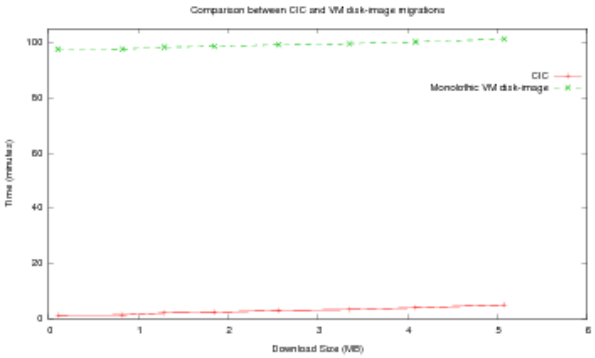
During the monolithic VM disk-image migration, the transferring includes both the VM disk-image part, containing the OS, and the other one containing the stored user data. In the case of our CIC migration, since the OS is stored on the Composable Block (CVM-image) which will be "cloned" on the destination cloud, only the User Data Block has to be transferred over the WAN. This specific performance analysis, since the destination cloud already holds all the necessary VM-CMPs to correctly pursue the CIC migration, represents the Best Case, as described in Section IV-B.

The transfer of the user data (the specific part of the monolithic VM disk-image or the User Data Block in the CIC methodology), allows to migrate the whole set of services running on the VM. During our two tests, we installed on the VMs a simple service which executes a download from a public web site: in this scenario, the user data size relies on the amount of bytes (MB) already downloaded. This means that, when the VM migration begins, in order to continue the download process on the destination cloud, the user data has to be transferred over the network. The time needed to transfer the user data depends on the number of bytes already downloaded when the migration begins: all the graphs depicted in Figure 5, show the time needed to perform the migration according to the two above mentioned approaches, and considering different sizes of the user data at the beginning of the migration phase.

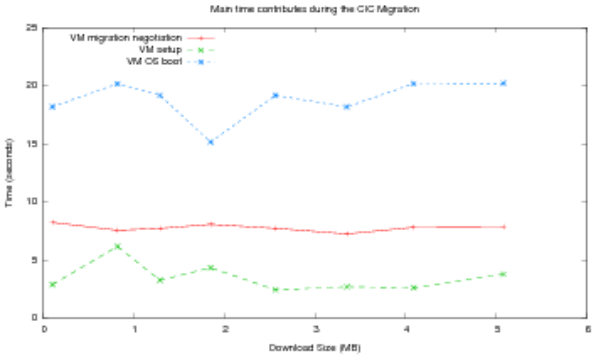
Graph 5(a) reports a comparison between the whole time (expressed in minutes on the y-axis) required to perform the VM migration. As shown in the graph, the time related to the migration of a monolithic VM disk-image whose OS part is 125 MB, depends on the number of bytes (expressed in MBytes on the x-axis) constituting the user data part when the migration starts. Regarding the CIC migration, the graph demonstrates how the total time required to accomplish the process, grows with respect to the size of the user data at the migration beginning. As pointed out, the time required in the first case in greater than the one necessary in the CIC approach.

Graph 5(b) is strictly related to the CIC solution: it represents the three main temporal contributions (expressed in seconds on the y-axis) involved in the cloning operation of a CVM-image. The graphs show how these contributions are constant and independent from the size (expressed in MBytes on the x-axis) of the user data (i.e. the User Data Block) when the CIC migration starts. More specifically, such contributions are: the time required to negotiate the migration on cloud B; the time needed to setup a resource (VM) on the cloud B; the time needed to boot the OS on the cloned VM in the destination cloud B.

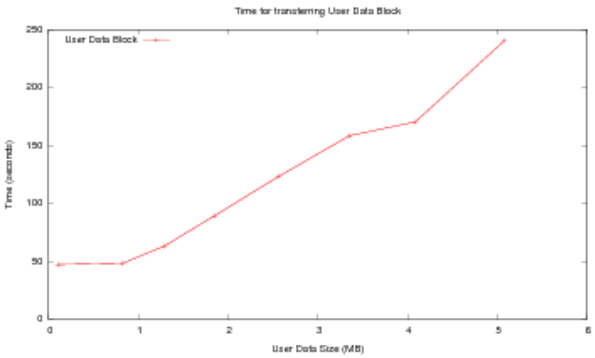
Graph 5(c) represents the time needed to transfer the User Data Block between cloud A and B when the CIC



(a) Comparison between the two different migration approaches



(b) Time needed to perform the three main phases of the CIC migration



(c) Time needed for transferring the User Data Block

Figure 5. Representation of the experimental time values

migration is employed. Such time values, together with the ones reported in the graph 5(b), determine the trend already depicted in the graph 5(a).

VI. Conclusions and Future Works

Currently, the worldwide cloud computing scenario includes several independent cloud providers, but the last trend is the cloud federation, where providers will be able to exchange and share virtual resources with each other. In this paper, we tackled the disk-image relocation, indispensable for both the"hot" and "cold" migration of virtual machines

in a federated cloud environment spread over WAN, where resources and enterprise assets can be provisioned, composed, and moved between different cloud providers. In order to reduce the cost of the VM disk-image relocation, we described and tested a new Composed Image Cloning (CIC) methodology. Our analysis has shown as the proposed approach scale well in distributed federated cloud scenarios, where virtual machine movement may occur frequently. Furthermore, an implementation practice, using and extending the OpenQRM platform, also providing experimental results has been proposed. In future, we plan to study the performance of our solution considering a large-scale federated cloud environment, evaluating the costs due to the VM disk-image relocation, employing a simulated environment including hundreds of clouds which dynamically perform VM migrations.

ACKNOWLEDGEMENTS

The research leading to the results presented in this paper has received funding from the European Union's seventh framework programme (FP7 2007-2013) Project RESERVOIR under grant agreement number 215605.

The authors would like to thank Dr. Alessio Di Pietro for his support in the development of the present work.

REFERENCES

[1] I. Foster, Y. Zhao, I. Raicu, and S. Lu, "Cloud computing and grid computing 360-degree compared," in *Grid Computing Environments Workshop, 2008. GCE '08*, pp. 1–10, 2008.

[2] Xen, "the powerful open source industry standard for virtualization", http://xen.org/.

[3] Kernel Based Virtual Machine, http://www.linux-kvm.org.

[4] VMware, "the powerful open source industry standard for virtualization", http://www.vmware.com/.

[5] C. Clark, K. Fraser, S. Hand, J. G. Hansen, E. Jul, C. Limpach, I. Pratt, and A. Warfield, "Live migration of virtual machines," in *ACM/USENIX Symposium on Networked Systems Design and Implementation*, 2005.

[6] M. R. Hines and K. Gopalan, "Post-copy based live virtual machine migration using adaptive pre-paging and dynamic self-ballooning," in *ACM/Usenix International Conference On Virtual Execution Environments*, pp. 51–60, 2009.

[7] W. Voorsluys, J. Broberg, S. Venugopal, and R. Buyya, "Cost of virtual machine live migration in clouds: A performance evaluation," in *CloudCom*, pp. 254–265, 2009.

[8] M. Andreolini, S. Casolari, M. Colajanni, and M. Messori, "Dynamic load management of virtual machines in a cloud architecture," in *ICST CLOUDCOMP*, 2009.

[9] F. Travostino, P. Daspit, L. Gommans, C. Jog, C. de Laat, J. Mambretti, I. Monga, B. van Oudenaarde, S. Raghunath, and P. Y. Wang, "Seamless live migration of virtual machines over the man/wan," in *Future Generation Computer Systems*, pp. 901–907, 2006.

[10] T. Hirofuchi, H. Ogawa, H. Nakada, S. Itoh, and S. Sekiguchi, "A live storage migration mechanism over wan for relocatable virtual machine services on clouds," in *The 9th IEEE/ACM International Symposium on Cluster Computing and the Grid*, 2009.

[11] T. Hirofuchi, H. Ogawa, H. Nakada, S. Itoh, and S. Sekiguchi, "A live storage migration mechanism over wan and its performance evaluation," in *The 3rd international workshop on Virtualization technologies in distributed computing*, pp. 67–74, 2009.

[12] OpenQRM, "the next generation, open-source Data-center management platform", http://www.openqrm.com/.

[13] Extensible Messaging and Presence Protocol (XMPP), http://xmpp.org/.