Kget+: An efficient tool for managing VM image snapshots

Jérôme Gallard

RenPar'19 / SympA'13 / CFSE'7
Toulouse, France, du 9 au 11 septembre 2009

Kget+ : An efficient tool for managing VM image snapshots

Jérôme Gallard*

IRISA, INRIA – Centre Rennes Bretagne Atlantique, Rennes, France.
Jerome.Gallard@inria.fr

Résumé
Virtualization technologies have recently gained a lot of interest in Grid computing as they allow flexible resource management. However, designing and implementing a complete framework to build VM images, to deploy and configure them automatically and to manage them (with migration and snapshotting operations) remains a real challenge. In this paper, after enumerating these challenges, we focus on the VM image snapshot management issue and present our tool, called `Kget+`, able to manage VM image snapshots in a very efficient way.

Mots-clés : Virtualization, Grid Resource Management, Efficient VM Image Copy, `Kget+`.

1. Introduction

Virtualization technologies provide flexible and powerful execution environments, offering isolation and security mechanisms, customization and encapsulation of entire application environments. Moreover, thanks to their suspend/resume and migration capabilities, they allow the bare hardware to be strongly decoupled from the system software, which is a predominant feature in the context of distributed architectures such as grids, which are exploited concurrently by multiple users. In this particular context, VMs can be exploited to encapsulate jobs and then make grid resource management easier. The challenge of managing applications on grids is moved to the problem of managing VMs on grids.
The rest of this paper is organized as follows : Section 2 introduces issues implied by the use of VM technologies in grids and presents more precisely the problem of VM snapshot management. In Section 3 we present `Kget+`, a tool able to manage VM image snapshots in an efficient way. Section 4 concludes and gives some perspectives of work.

2. Issues implied by the use of VM technologies in grids

Designing and implementing a framework able to manage user jobs in VMs at grid level requires to consider each step of a common job submission in a grid context : (i) VM image creation, (ii) VM image management (iii) job submission, VM deployment and job starting, (iv) VM/job life cycle (VM migration, suspend/resume, snapshots, ...) and finally (v) job completion and VM shutdown.
In this document, we focus on the VM storage management issue and more precisely, on the VM image snapshot issue by proposing a tool, called `Kget+`, able to copy efficiently VM image snapshots from n nodes to one frontend (more information about the other issues could be found in [1]).

The VM image snapshot management issue

One drawback of the use of VMs in grids and clusters relates to the VM image snapshot management. There are several methods to manage VM images (on each local node, on a remote server, ...), each method having their own good and bad points. In this paper we focus on the remote server method at cluster level by considering the copy of n VM image snapshots from n physical nodes to a frontend which implies a network bottleneck at the frontend side. To solve this problem, we have designed and implemented `Kget+`, which is presented in the next Section.

* The INRIA team carries out this research work in the framework of the XtreemOS project partially funded by the European Commission under contract #FP6-033576.

3. `Kget+` : Design and Experiments

`Kget+` is our prototype allowing to copy VM image snapshots from n nodes to one frontend in an efficient way. It follows two principles : (i) to regulate the network bandwidth from the frontend point of view, (ii) to free a large part of the set of nodes in a minimum of time.

To do this, `Kget+` makes sure that at all times one and only one node sends a VM image to the frontend (principle (i)). Parallelly, other nodes coordinate among themselves to group all VM images on a smaller number of nodes (principle (ii)). In this way, the network between all the nodes is used parallelly and the result is that several nodes are freed very quickly and the frontend network is never saturated. Of course this solution assumes that there is enough available disk space on the nodes to make the copies.

To do this, we propose the following algorithm :

While there is a node waiting to transfer a VM snapshot to the frontend :

- *copy one VM snapshot from the node containing the lowest number of VM snapshots*[2] *to the frontend.*
- *from each other node, copy VM snapshots to another node except if :*
  - *the destination node is already sending or receiving a VM snapshot,*
  - *the destination node has less VM snapshots than the sender node,*
  - *the sender node is already sending or receiving a VM snapshot.*

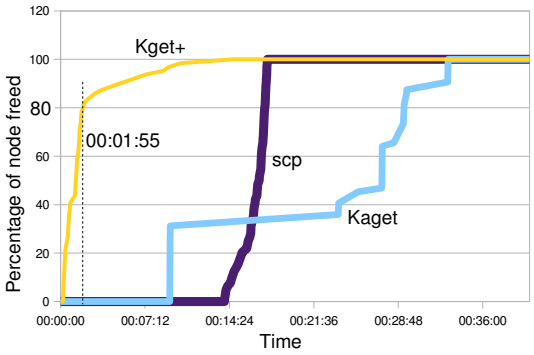

FIG. 1 – Comparison between multiple `scp`, `kaget` and `Kget+`. `Kget+` allows to freed 80% of the nodes in less than 2 minutes.

After implementing this algorithm, we made some evaluations of it on the Paravent cluster from Rennes Grid'5000 site[3] [2]. In our experiments we launch a job composed of 64 VM images on 64 physical nodes and we measure the time needed to copy the VM image snapshots from all nodes to the frontend with multiple simultaneous `scp`, with `Kaget`[4] and with our tool `Kget+`. The size of one VM image snapshot is approximately 512MBytes. Figure 1 presents the percentage of nodes freed (Y-axis) according to the elapsed time (X-axis). This figure shows that, thanks to `Kget+`, a good scheduling of the copies gives very efficient results. Indeed, with `Kget+` all nodes are freed in less than 15 minutes whereas with multiple `scp` and `Kaget` all nodes are freed in less than 18 and 34 minutes respectively. In addition, with `Kget+` 80% of the nodes are freed in less than 2 minutes (6% of the wall time).

4. Conclusion and Future Work

In this paper we present the design of `Kget+`, our prototype able to copy VM image snapshots from n physical nodes to one frontend in an efficient way. In addition we give results of preliminary experiments. This tool is able to schedule the copy of all the VM image snapshots from all the nodes to one frontend with the goal of : (i) regulating the network bandwidth from the frontend point of view, (ii) freeing a large part of the set of nodes in a minimum of time. This tool should be integrated with `VMdeploy`[5] our VM management prototype able to deploy and manage VMs at grid level. This one needs improvements, however, it is a good framework to study one by one, all the issues implied by the use of VMs in grids which, from our point of view, will pave the way to scientific clouds.

Bibliographie

1. Jérôme Gallard et al. VMdeploy, Improving Best-Effort Job Management in Grid5000. Technical Report RR-6764, INRIA, December 2008.
2. Raphaël Bolze et al. Grid'5000 : A Large Scale and Highly Reconfigurable Experimental Grid Testbed. *International Journal of High Performance Computing Applications*, 20(4) :481–494, 2006.

[2] This could be easily extended to consider the whole size of VMs instead of the number.
[3] https ://www.grid5000.fr
[4] http ://taktuk.gforge.inria.fr/kanif/
[5] https ://www.grid5000.fr/mediawiki/index.php/VMdeploy