Snooze: an Autonomic and Energy-Efficient Management System for Private Clouds

Matthieu Simonin, Eugen Feller, Anne-Cécile Orgerie, Yvon Jégou, Christine Morin

Snooze: an Autonomic and Energy-Efficient Management System for Private Clouds

Matthieu Simonin[1], Eugen Feller[1], Anne-Cécile Orgerie[2], Yvon Jégou[1] and Christine Morin[1]

[1] INRIA, Myriads team, IRISA, Rennes - France
{matthieu.simonin, eugen.feller, yvon.jegou, christine.morin}@inria.fr
[2] CNRS, Myriads team, IRISA, Rennes - France
anne-cecile.orgerie@irisa.fr

Abstract. Snooze is an open-source scalable, autonomic, and energy-efficient virtual machine (VM) management framework for private clouds. It allows users to build compute infrastructures from virtualized resources. Particularly, once installed and configured it allows its users to submit and control the life-cycle of a large number of VMs. For scalability, the system relies on a self-organizing and healing hierarchical architecture. Moreover, it performs energy-efficient distributed VM management. Therefore, it implements features to monitor and estimate VM resource utilization (CPU, memory, network Rx, network Tx), detect and resolve overload/underload situations, perform dynamic VM consolidation through live migration, and finally power management to save energy. Last but not least, it integrates a generic scheduler which allows to implement any VM placement algorithm. This demo will expose the Snooze's main properties (scalability, energy-efficiency, autonomy, and fault-tolerance) through its graphical interface.

Keywords: Cloud computing, virtualized clusters, energy-efficiency, autonomic resource management

1 Introduction

The ever growing appetite of new applications for Cloud resources leads to an unprecedented electricity bill for providers. More than the financial cost, the environmental cost is worrying as it threatens the Cloud expansion. It has indeed become a major brake to the deployment of new infrastructures. For instance, in 2010, Google used about 900,000 servers that consumed around 2 billions kWh [1]. One can wonder whether all this energy was necessary. Indeed, Cloud systems are often facing highly variable loads, but they are sized to face the peak power load and their resources are most of the time powered-on even when idle mainly for the sake of reactivity [2]. As current server architectures are not power proportional (i.e. their power consumption is not proportional to the load they are facing) [3], and as their idle consumption (i.e. their power consumption when they are on but idle) is high compared to their peak power consumption, huge amounts of energy are lost when powered-on servers are idle and this happens often.

In this demo, we will present Snooze [4, 5], an autonomic and energy-efficient management system for private clouds. Benefiting from its hierarchical self-configuring architecture, Snooze embeds different autonomic energy-efficient management mechanisms:

- switch off of idle servers,
- energy-efficient VM placement,
- server underload detection and mitigation using live migration to unbalance the load and to increase the number of unused resources,
- periodic VM consolidation.

2 Snooze Architecture

Previous work has shown that hierarchical architectures can greatly contribute to system scalability [6]. Following this principle of splitting the knowledge among independent managers, Snooze is based on an autonomic hierarchical architecture shown in Figure 1.

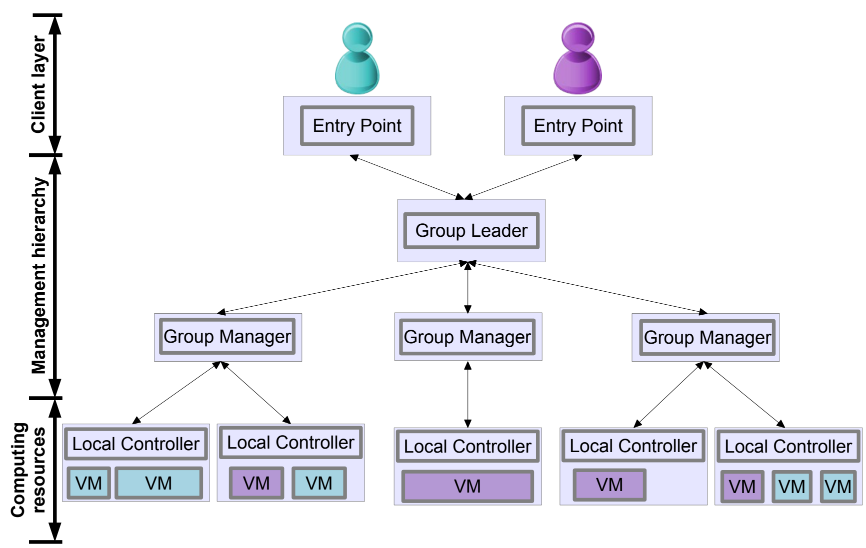

Fig. 1. Snooze architecture

Each server (i.e. physical node) is managed by a *Local Controller* (LC) which monitors the VMs of the node and enforces the VM life-cycle and node management commands coming from higher levels of the hierarchy. A *Group Manager* (GM) is in charge of managing a subset of LCs. It integrates energy-management mechanisms such as overload / underload mitigation and VM consolidation policies. A *Group Leader* (GL) manages the GMs, it assigns the LCs to GMs, and it deals with clients VM submission requests.

On Figure 1, servers are represented by light gray rectangles. On this example, we chose to have three GMs but this number is defined by the administrator depending on the desired level of redundancy (for fault-tolerance and performance). The main roles and associated functionalities of each component of Snooze's hierarchy are summarized in Table 1.

Components	**Roles**	**Functionalities**
Entry point	user interface	availability
Group Leader	deals with client requests assigns GM to LCs dispatches VMs among GM	load-balancing
Group Manager	places VMs overload/underload mitigation VM consolidation (migration plan)	scalability self-optimization energy-efficiency
Local Controller	monitors VMs enforces VM states (start, migration, ...) power off resources overload/underload detection	self-configuration fault-tolerance
Virtual Machines	run applications	user-friendly

Table 1. Overview of Snooze hierarchy components

Snooze is written in Java. It currently comprises 15,000 lines of highly modular code. It is distributed in open-source under the GPL v2 license [7]. It relies on the *libvirt* library which provides a uniform interface to most of the modern hypervisors [8].

3 Demonstration Scenarios

Snooze has been validated on Grid'5000, the French experimental test-bed to support experiment-driven research in all areas of computer science related to parallel, large-scale or distributed computing [9]. Grid'5000 comprises 6,500 cores geographically distributed in 10 sites linked with a dedicated gigabit network. This demo will be performed on this platform.

In particular in this demo, we will show the main properties of Snooze: the periodic VM consolidation, the power down of unused nodes and their boot when they are needed, the deployment of VM over distant sites, the reconfiguration, fault-tolerance and self-healing mechanisms. Screencasts of the graphical user interface (GUI) of Snooze have been made to illustrate this demo and will be utilized in case of network connection issues during the live demo. The Snooze GUI shows the hierarchy in real-time (with a configurable refreshment period).

During this demo, the following scenarios will be presented:

– a typical Snooze deployment using 50 nodes on 3 different sites of Grid'5000 (and thus on heterogeneous nodes);
– the Snooze energy management in action shutting down unused nodes;
– the wake-up of sleeping nodes when new VMs need to be deployed;
– the on-demand migration mechanism of all the VMs managed by one local controller;
– the periodic VM consolidation for energy saving purposes which migrates VMs among distant sites;
– the self-healing mechanisms and autonomic reconfiguration when a Group Manager or the Group Leader crashes.

4 Conclusion

Snooze [4, 5] is a scalable, autonomic, and energy-aware virtual machine management framework. Its name stands for "take a nap" and was selected to emphasize the energy management features of the system, such as the autonomic powering off policy for unused resources.

Snooze [7] is open-source and can thus be used by any research team dealing with Cloud computing, energy efficiency, and resource management in large-scale distributed systems. This demo will also point out the easiness of deployment and management of virtualized clusters provided by Snooze.

References

1. Koomey, J.: Growth in data center electricity use 2005 to 2010. Technical report, Analytics Press (2011)
2. The Green Grid: Unused servers survey results analysis. Technical report (2010)
3. Barroso, L., Holzle, U.: The Case for Energy-Proportional Computing. Computer **40**(12) (2007) 33–37
4. Feller, E., Rohr, C., Margery, D., Morin, C.: Energy Management in IaaS Clouds: A Holistic Approach. In: CLOUD: IEEE International Conference on Cloud Computing. (2012)
5. Feller, E., Rilling, L., Morin, C.: Snooze : A Scalable and Autonomic Virtual Machine Management Framework for Private Clouds. In: CCGrid: IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing. (2012)
6. Perera, S., D., G.: Enforcing user-defined management logic in large scale systems. In: IEEE Congress on Services. (2009) 243–250
7. Snooze: http://snooze.inria.fr. GPL v2 licence (2012)
8. Red Hat: libvirt: The virtualization API. http://libvirt.org/ (2012)
9. Bolze, R., Cappello, F., Caron, E., Daydé, M., Desprez, F., Jeannot, E., Jégou, Y., Lanteri, S., Leduc, J., Melab, N., Mornet, G., Namyst, R., Primet, P., Quetier, B., Richard, O., El-Ghazali, T., Touche, I.: Grid'5000: A Large Scale And Highly Reconfigurable Experimental Grid Testbed. International Journal of High Performance Computing Applications **20**(4) (2006) 481–494