Octopus: self-service cloud of virtual machines

Antonio Cisternino (cisterni@di.unipi.it), Maurizio Davini (mau@df.unipi.it), Marco Mura (mura@di.unipi.it)

IT Center, Università di Pisa, L.go B. Pontecorvo 3, 56127 Pisa

Virtual machines are a standard unit of resource allocation unit for cloud environments, in fact, VMs are a building block of the popular Amazon EC2 public cloud and even Microsoft introduced in Azure the VMRole providing a similar feature. Virtualization has been also used for implementing cloud infrastructures because of its natural ability to decouple the physical hardware from logical servers.

The increasing density of computational power allows packing a significant number of virtual machines on a single node, leading to an exponential growth of the number of logical servers to be managed by the cloud infrastructure. To investigate smart policies to administer such a large number of virtual machines we have developed Octopus, a lightweight system for scheduling virtual machines on a cluster of hypervisors. This work, initially presented at Cloud Futures 2010 workshop held in Microsoft Research, Redmond, is still under development even though modern cloud stacks feature the same architectural.

Octopus has been implemented using Microsoft Hyper-V in order to exploit the WMI interface to control the hypervisor programmatically with the F# programming language. The original goal was to design a system capable of moving virtual machines across different computing nodes in order to optimize the workload and pack computations to save energy by turning off nodes. Moreover the VM creation is under control of the final user through a Web page where it is possible to ask for specific requirements about computing cores, memory and the OS image to be provisioned. All VMs are subject to explicit renewal to ensure that resources are not wasted and a set of policies set by the administrator govern resource management.

More recently, our investigation has focused on the possibility of using expert systems to express complex policies and to govern this ever increasing set of virtual machines. For this reason, we have embedded the CLIPS expert system inside Octopus in order to rely on a full rule-based expert system engine to define the resource management policies: the Octopus code asserts facts about VMs in the CLIPS systems and rules access system primitives exposed as functions invoked by triggered rules. The well-known RETE algorithm ensures a fast execution of policy while ensuring the ability to define policies that may even contain conflicting rules.

In order to reduce resource-wasting Octopus heavily uses differencing disks allowing creating an instance of an operating system with a very small disk overhead (under 100Mb for a new OS instance). This helps migration of VMs across different storages if live migration is not supported by a particular hypervisor.

During our tests using the Octopus system we have found that x86 is a reasonable contract between the cloud service provider and its users: it clearly features a general purpose API that is accessible and flexible at the cost of memory overhead due to multiple copies of the OS code for different VMs. We have also noticed that sometimes virtualization introduces a speedup rather than a slowdown in software execution since not always software is capable to exploit all available resources; in this case, virtual machines offer configurations closer to the hardware used as reference for some programs.

We believe that a cloud for research should be based on VMs rather than APIs and software stacks, so it provides a flexible environment where resourced can be offered in a self-service fashion, offering the service to a larger research audience than of the traditional HPC. Octopus is a research prototype, but open and commercial cloud stacks offer the primitives to build such infrastructure. Last, but not least, a cloud of VMs would also be the ideal foundation for supporting a VDI infrastructure.