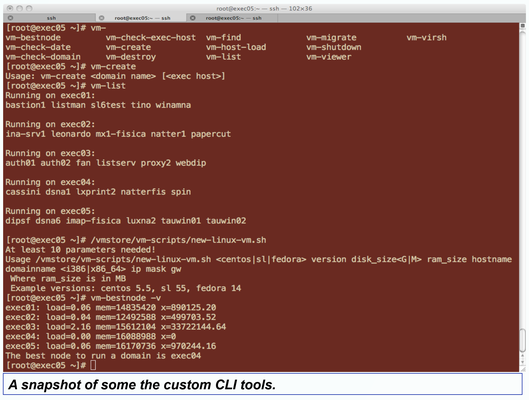
Design and implementation of a reliable and cost-effective cloud computing infrastructure: the INFN Napoli experience

[a,b]Vincenzo Capone, [b]Rosario Esposito, [b]Silvio Pardi, [b,c]Francesco Taurino, [b]Gennaro Tortone

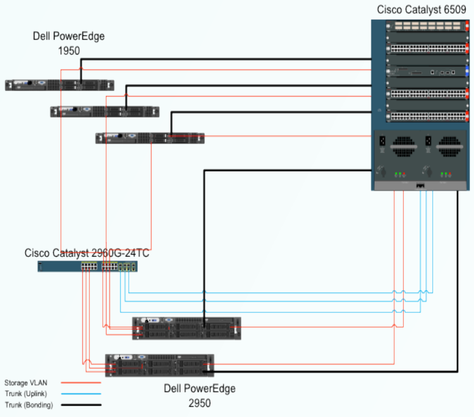
[a] Università degli Studi di Napoli Federico II – Napoli, Italy
[b]INFN-Napoli -Campus di M.S.Angelo Via Cinthia– 80126, Napoli, Italy
[c]CNR-SPIN - Campus di M.S.Angelo Via Cinthia– 80126, Napoli, Italy
Email: ecapone@na.infn.it, resposit@na.infn.it, spardi@na.infn.it, taurino@na.infn.it, tortone@na.infn.it

Introduction

In this work, we describe an IaaS (Infrastructure as a Service) cloud computing system, with high availability and redundancy features which is currently in production at INFN-Napoli and ATLAS Tier-2 data centre.
The main goal we intended to achieve was a simplified method to manage our computing resources and deliver reliable user services, reusing existing hardware without incurring heavy costs. A combined usage of virtualization and clustering technologies allowed us to consolidate our services on a small number of physical machines, reducing electric power costs. As a results of our efforts we developed a complete solution for data and computing centers that can be easily replicated using commodity hardware.



A snapshot of some the custom CLI tools.



Hardware

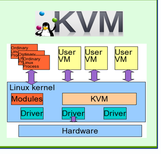
We started from commodity hardware we already owned, that was upgraded in order to fulfill the requested performance. In particular, three Dell PowerEdge 1950 rack servers have been used as VM executors, and two Dell PowerEdge 2950 as VM stores.
All servers are equipped with dual Intel Xeon E5430, providing 8 cpu cores per server, with 8 Gbyte of RAM. The upgrades consisted in a 8 Gbyte RAM and 2 ports Ethernet NIC on hypervisors, 6 x 1.5 TByte SATA hard disk and a 4 ports Ethernet NIC on both storage servers. The storage server disks are configured in RAID5 (dm-raid software mode), so the total available storage space is 7.5 Tbyte per server. Hypervisor servers have 2 x 500 Gbyte disks configured in RAID1. Furthermore a dedicated 24 gigabit ports Cisco Catalyst 2960G switch was added to the hardware configuration to provide a dedicated storage LAN.

Network

The main requirements for the network serving our infrastructure are: performance, reliability and resiliency. To achieve these goals, we set up a double path between every hypervisor and both storage servers, with two different switches involved, so that a failure in one of them doesn't impact on the execution of the Virtual Machines, whose disk images are hosted on the storage servers. The Cisco Catalyst 6509 is the core switch of our science department network infrastructure, and every server is connected to it via the onboard dual gigabit ethernet port, in LACP bonding mode, so to provide the necessary connectivity and the sufficient bandwidth to the VMs: this link is in trunk mode, so that every VM can be connected to the desired VLAN. The second switch (Cisco 2960G) is connected to the former via a 3 x 1 Gbit LACP bond link. A private VLAN hosts the data traffic between the storage servers and the hypervisors; within this VLAN every storage server is connected with three gigabit links to the Cisco 2960G and the fourth to the Cisco 6509, while every hypervisor is connected with one link to both switches; the multiple connection of the servers to the two switches is achieved with the Adaptive Load Balancing mode. Within this topology, the Cisco 2960G is completely dedicated to the network traffic of the storage VLAN, while the Cisco 6509 is used as access switch towards the LAN, and as the redundant switch for the storage VLAN.

Software

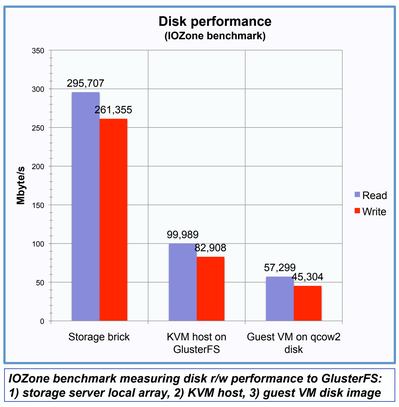
The OS used on all servers was initially Scientific Linux 5.5, with KVM as virtualization system. We selected KVM as the best architecture for virtualization on modern processors with fast hardware virtualization support (VT-x and NPT on Intel or AMD-V and EPT on AMD). After, we updated all servers to Scientific Linux 6.2 to use the new KVM version and KSM (Kernel Samepage Merging), a memory deduplication feature which enables more guests sharing to share the same memory pages of the host.



Storage

We chose GlusterFS as a fault-tolerant backend storage for virtual machine images. GlusterFS is an open source, clustered file-system for scaling the storage capacity of many servers to several petabytes. It aggregates various storage servers or bricks over Infiniband RDMA and/or TCP/IP interconnection into one large parallel network file system. Key features of GlusterFS:
- Modular, stackable storage OS architecture
- Data stored in native formats
- No metadata – Elastic hashing
- Automatic file replication with self-healing.

In the GlusterFS world a *volume* is a logical collection of bricks, where each brick is an export directory on a server in the trusted storage pool. Most of the Gluster management operations happen on the volume. In our local setup we used a replicated Gluster volume created on top of 2 servers to store virtual machine disk images in qcow2 file format. Each storage server exports a storage brick which consists of an ext3 file system built on a Linux software RAID5 array.
In the picture on the right are shown the disk performances under various use cases.



IOZone benchmark measuring disk r/w performance to GlusterFS: 1) storage server local array, 2) KVM host, 3) guest VM disk image

Features

We have developed some CLI scripts for day by day tasks on our private cloud in order to reduce administration efforts, like rapid provisioning of guest, listing, rapid migration, load balancing and automatic migration and restart of VMs hosted on a failed hypervisor.
With our deployment we've achieved all the goals we intended to: ease of management, high availability and fault tolerance. The functional integrity of the whole cloud system is preserved even after the fault of multiple elements of the system: in fact, no other effects, but the declining of the overall performance, happens after the failure of one of the two switches, one of the two storage servers, all but one of the KVM hypervisors, even if all this happens at the same time.
In conclusion, our system has proved itself a solid and efficient solution, after more than one year of uninterrupted uptime, to deploy all those services that don't require a heavy load on the I/O subsystem, but that are a crucial element of a modern datacenter.