VIDA – a virtualized data processing and analysis facility

Frederik Orellana[1],...
October 2009

1 Niels Bohr Institute, University of Copenhagen

Introduction

This document describes a data processing facility, VIDA, that takes advantage of virtualization technology to attempt to provide a unified experience for scientists doing computationally intensive research.

The work was motivated by a desire to streamline and simplify the steps a researcher has to go through in order to use a distributed infrastructure to process or simulate larger amounts of data. Specifically, the aim is to prepare for analyzing the data that will now very soon arrive from the Large Hadron Collider at CERN.

It should be noted that we are restricting ourselves to serial or pleasingly parallel computations and that with larger amounts of data we mean amounts below those involved in typical official CERN production runs: the system is designed primarily for usability and not for efficiency. Really large production runs should be carried out on the grid infrastructure that was designed for this.

The main idea is to strictly support the way many researchers do their computational work: 1) log in to a work station and compile, debug and test a given application, 2) run this same application multiple times[2], either with different parameters or with different input files. Typically 2 is done via some sort of batch or queuing system.

The crucial point here is that in order to painlessly go from step 1 to step 2, exactly the same environment (operating system, installed software and libraries) should be present both on the workstation where the development and initial testing was done *and* on the machine where the jobs are later placed by the mentioned batch or queuing system.

The way we have chosen to address this is via virtualization and a catalog of disk images. With a disk image, we here refer to a disk image that provide an operating system and can be booted by a virtual machine hypervisor like Xen.

More detailed, the envisioned working procedure of a researcher on our facility is then:

1. Select, boot up and log in to a virtual machine – this is a one-step procedure, initiated by simply logging in via SSH to the front-end machine - where-after the researcher is prompted to choose a virtual machine.
2. Compile, debug, test.
3. If necessary, save the current state of the virtual machine as a new image and add it to the catalog.[3]
4. Prepare and run multiple jobs, each requiring to run in the same kind of virtual machine.

The implementation of the mentioned one-step procedure is a script, “cloud_init”, replacing the normal login shell, and an accompanying PAM module, “pam_confusa.so”. To manage jobs, we have developed a prototype GUI, GridPilot [GRIDPILOT].

Log-in

The front-end machine of the facility supports log-in via standard SSH with the user name “cloud”. On log-in, cloud_init uses the WAYF and Confusa web services to verify the user's user name and password by redirecting him to his home institution for log-in. After that, cloud_init uses the same web services for generating short-lived X.509 credentials that then reside on the front-end (in ~/.globus/).

2 referred to as running jobs.

3 This is not yet implemented.

After log-in on the front-end is completed, cloud_init prompts the user if he wants to be logged in on any virtual machines he may have running or if he wants to choose and boot up a fresh virtual machine. After he has made his choice, if needed, cloud_init boots up a virtual machine and mounts network storage partitions on behalf of the users and then finally logs the user in on the chosen virtual machine.

Virtualization, disk images and software provisioning

The script that is run when a user logs in on the front-end machine uses the OpenNebula CLI for finding out which machines the user already may have running, getting a list of available images and booting up a virtual machine using one of these images.

Some of the images in question are modified images from the CernVM project. These images include the cvmfs/fuse kernel module, which is used for providing software. For more detail on this, see [CERNVM]. Others simply have software pre-installed. All images are raw disk images and are used for booting up virtual machines via the Xen hypervisor.

Network storage

To allow mounting network storage inside the virtual machine on behalf of the user, the virtual machine must have the davfs2/fuse kernel module installed and loaded and the user must be member of a certain so-called virtual organization (see [GRIDDK]). All images in our catalog and all of our users satisfy these requirements.

Currently, all virtual machines mount a network partitions on /mnt/cloud/_name and the CernVM machines moreover mount a read-only network partition on /mnt/atlas_data/.

The storage is provided over HTTPS from the dCache installations operated by the NDGF.

[TODO: NDGF]

Job and file management

GridPilot is a graphical application, written in Java, that allows automating the preparation and control of large amounts of jobs. GridPilot moreover allows running jobs on other systems, including Amazon's EC2 and the NorduGrid and EGEE grids.

The application can be downloaded from the GridPilot web site and we distribute it to Danish academia with a configuration file customizing it for VIDA.

Use cases

ATLAS data processing

Statistical epistasis analysis

Variance at risk simulation

Outlook

--- add VM functionality to grid middleware [TODO: Anders?]

Bibliography

GRIDPILOT, F. Orellana, “GridPilot – a graphical front-end to distributed compute systems”, *in preparation*, http://www.gridpilot.dk/

GRIDDK, F. Orellana, J. Berthold, J. Bardino and B. Sedoc, “grid.dk – a compute infrastructure for Danish academia”, *in preparation*

CERNVM, P. Buncic, C. Aguado Sanchez, J. Bloomer, L. Franco, S. Klemer and P. Mato, “CernVM - a virtual appliance for LHC applications“, PoS(ACAT08)012, http://pos.sissa.it/cgi-bin/reader/conf.cgi?confid=70