A ground-up approach to High-Throughput Cloud Computing in High-Energy Physics

Dario Berzano

Supervisors: M. Masera, G. Ganis, S. Bagnasco

Università di Torino - Doctoral School of Sciences and Innovative Technologies
Virtualization and Cloud Computing in High-Energy Physics
Institutions and projects involved

- INFN Torino
  Centro di Calcolo

- Università di Torino

- CERN PH-SFT
  PROOF and CernVM

- The ALICE Experiment
  @ LHC
Computing in High-Energy Physics

From raw detector hits to formats suitable for repeated user analysis

- Massive data: **177 PB** in total (at LS1)
  - highest rate: **LHCb** (1 MHz trigger)
  - largest event size: **ALICE** (6 MB in Pb-Pb)
- High Throughput Computing:
  - network and storage set our limit
  - events are **independent**: trivial parallelism

Raw collected → ESD reconstructed → AOD filtered

- LHCb: 7 PB
- ATLAS: 55 PB
- CMS: 85 PB
- ALICE: 30 PB

Dario.Berzano@cern.ch - A grounds-up approach to High-Throughput Cloud Computing in High-Energy Physics
The Grid: many geographically distributed computing sites with a tiered structure

- Each site provides computing and storage
- Data is replicated: faster access and backup
- Tier boundaries are nowadays loose:
  - better network and higher CPU density
  - in practice they all talk to each other

Main data flow:

<table>
<thead>
<tr>
<th>Tier</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tier-0</td>
<td>CERN only</td>
</tr>
<tr>
<td>Tier-1</td>
<td>sim + rec</td>
</tr>
<tr>
<td>Tier-2</td>
<td>user analysis</td>
</tr>
<tr>
<td>Tier-3</td>
<td>local access</td>
</tr>
</tbody>
</table>

- 10,000 users
- 50,000,000 jobs
- 340,000 cores
- 180 PB of storage
Virtualization and Cloud Computing

Virtualization
*hardware resources do not correspond to exposed resources*

**Partitioning resources**
export virtual small disks or CPUs not corresponding to actual hardware resources (e.g. *Dropbox*)

**Emulating resources**
reprocess data from old experiments by emulating old hardware: *Long Term Data Preservation*

Cloud Computing
*leverage virtualization to turn computing resources into services*

**Software as a Service**
software is not installed but runs somewhere and exposes a remote access interface (e.g. *Gmail*)

**Infrastructure as a Service**
virtual machines run somewhere and they can be accessed as if they were physical nodes
Administrative domains in the cloud

**infrastructure**

virtual infrastructure administrator
configures virtual machines: does not care about the underlying hardware

**services**

user
uses the services just like before, completely unaware of virtualization

administrators of distributed and independent clouds
manage the hardware, replace disks when broken, monitor resources usage, coordinate “local” and “remote” users
Benefits of cloud computing

- Ensures a **consistent environment** for your software
- Clear **separation of administrative domains**
- Support different use cases on the same hardware infrastructure, and rapidly move resources between them: **multi-tenancy**
- **Opportunistic exploitation** of otherwise unused resources: a good example are **high-level trigger farms** outside of data taking
Drawbacks of cloud computing

- Virtualization exposes to the VMs the lowest common denominator of hardware features (such as CPU specs): no architecture-specific optimization possible
- However loss of performances is invisible in most cases:
  - near zero loss on CPU bound tasks
  - Grid jobs slowed down by remote I/O
- Virtualization is appropriate for some use cases (Grid-like) and not suitable for others (real-time applications, triggers, etc.)

Grid-like applications: cloud has more benefits than drawbacks
Building a private cloud from a Grid computing center
INFN Torino’s computing center

Fact sheet

- Gross storage: **1 000 TB**
- Computing: **14 kHS06**
- Job capacity: **1 100 cores**
- Network: **1/10 GbE**

Torino’s computing capacity evolution

- Classified as a Grid Tier-2
- Main customer: the ALICE experiment (> 90% jobs)

**Since Dec 2011: all new hardware configured for the cloud**

Dario.Berzano@cern.ch - A grounds-up approach to High-Throughput Cloud Computing in High-Energy Physics
The orchestrator: OpenNebula

- **Controls** and **monitors** hypervisors and other virtual resources
- Manages **lifecycle** of virtual machines and virtual farms
- **Repository** of base images and **marketplace** for external images
- Web interface with **virtual VM consoles**

- One of the **first tools** available
- Very **robust**
- **Open source**
- Easily **customizable** via the **API**
Design of our cloud architecture

Service hypervisors
*high availability for critical services*

- VMs run from a *shared disk*
- not suitable for high I/O
- live migration: move running VMs, zero service interruption
- e.g. *local Grid head node*

Working class hypervisors
*high performance applications*

- VMs run from the *local disk*
- optimized for high I/O
- failures acceptable: expendable VMs, jobs resubmitted
- e.g. *Grid worker nodes*

Shared storage for services

- 2 redundant GlusterFS servers
- Self-recovery upon disk failures

Constraints

- Integration with non-cloud part
- Progressive migration to cloud
Techniques for fast deployment

Base VM image with the OS: the “root disk”

- **QCoW2**: image grows on use
- **Slow writes**: use ephemeral disk for that
- Popular images (e.g. Grid nodes) cached and booted as snapshots

Extra space: the “ephemeral storage”

- Raw disk (no QCoW2) with `fallocate`: $O(1)$
- Standard `ext3/4` creation is $O(size)$: use XFS or “lazy” `ext4` init: $O(1)$

Images cache

- Delete/create img in OpenNebula as usual
- Single sync cmd
- Fast torrent-like sync: all nodes are seeds

Deployment times

- 1 node: < 15 s
- 40 nodes: < 2 min
Multiple tenants and the Grid

The Grid is just a special tenant of our private cloud

- **Objective:** computing resources never wasted
  - the **Grid** always needs resources
  - relinquishes resources in favor of others
- It may take **hours** for our Grid VMs to drain
  - of 1,000 jobs, ~75 **finish** per hour
  - our Grid VMs currently have **6 cores**
- Little free space for small use cases (~50 slots): larger use cases should book resources “the evening before”

Draining Grid virtual machines

- **Command** to set VMs to “drain mode”
- **No new jobs** are accepted
- OpenNebula **shutoff** when all jobs done
Sandboxed virtual farms

New resources **provisioning model**
- Each tenant has some resources **quota**
- **Base images** for various OSes provided

Per-tenant **sandboxed** private networks
- Isolated at MAC addr level via **ebtables**
- User can map a public **Elastic IP** to any VM

Access via standard **Amazon EC2** commands
- **Clearer** to use than native OpenNebula’s
- If changing from OpenNebula to other tools, **still same commands**
- **EC2 clients** ready to use on **public login nodes**

**Under the hood**
- **Automated** creation of sandboxed env: user, quota, network
- Networking isolation and Elastic IPs via **Virtual Routers**
- **Virtual Routers auto launched** at creation of sandbox
Virtual router: OpenWRT

Each sandboxed network is managed by a Virtual Router

- Tiny: 1 CPU, < 200 MB RAM, runs on Service Hypervisors
- Has Public and Private IP: provides DNS, DHCP, NAT, firewall
- Elastic IP functionality obtained via port forwarding
- Invisible and inaccessible to users

A customized OpenWRT
- Linux distro for domestic routers
- Web management interface
- "Network as a Service"
From Grid to clouds

• By exposing a standard API (EC2), Torino’s private cloud is prepared for a future of interoperating small clouds

• Fully configured VMs submitted both at user and experiment level

• Users can keep using the same workflow (e.g. Grid submission) and have in addition the possibility of launching VMs directly

• Very generic requirements for local cloud sites

• Experiments can centrally ensure environment consistency in a non-invasive manner for the local sites

Clouds provide genuine abstraction of resources (CPU, RAM, disk...)

Dario.Berzano@cern.ch - A grounds-up approach to High-Throughput Cloud Computing in High-Energy Physics
Running elastic applications: the Virtual Analysis Facility
Cloud-awareness

Clouds can be a troubled environment

- Resources are diverse
  → Like the Grid but at virtual machine level
- Virtual machines are volatile
  → Might appear and disappear without notice

Building a cloud aware application for HEP

- Scale promptly when resources vary
  → No prior assignment of data to the workers
- Deal smoothly with crashes
  → Automatic failover and clear recovery procedures

Usual Grid workflow → static job pre-splitting ≠ cloud-aware
PROOF is cloud-aware

PROOF: the Parallel ROOT Facility

• Based on unique advanced features of ROOT
• Event-based parallelism
• Automatic merging and display of results
• Runs on batch systems and Grid with PROOF on Demand

PROOF is interactive

• Constant control and feedback of attached resources
• Data is not preassigned to the workers → pull scheduler
• New workers dynamically attached to a running process

**Interactivity is what makes PROOF cloud-aware**
PROOF dynamic scheduling

Adaptive workload: very granular pull scheduler

Nonuniform workload distribution

Packet generator

master

worker

get next

ready

process

packet

time

get next

ready

process

packet

time

get next

ready

process

Packet per worker

Events

0 500 1000 1500 2000 2500 3000 3500 4000

Events per worker

Worker activity start (seconds)

Mean

RMS

Query Processing Time (s)

Worker activity stop (seconds)

Mean

RMS

all workers are done in ~20 s

Uniform completion time

Dario.Bezzano@cern.ch - A grounds-up approach to High-Throughput Cloud Computing in High-Energy Physics
PROOF dynamic workers

User workflow

1. User queues N workers
2. Wait until 1 worker is up
3. User launches analysis
4. Workers gradually join

Under the hood

- Initially available bulk init
- init process
- master initiate worker
- new workers autoregister deferred init
- process init
- init process
- init worker
- register init
Scheduling PROOF workers on the Grid

Rationale

• Show how resources ramp up on Grid sites according to their size
• Compare time-to-results of the same analysis with PROOF and Grid

Benchmark conditions
ATLAS Italian Grid sites + CERN
100 jobs queued

Obtained jobs can be used as:
PROOF workers pull scheduling: dynamic workers and workload
Grid jobs push scheduling: wait for the last job to complete

Large sites give resources more rapidly

Dario.Berzano@cern.ch - A grounds-up approach to High-Throughput Cloud Computing in High-Energy Physics
**PROOF vs. Grid scheduling**

- **Time to results**: effective time to wait before having the results.

- **Batch jobs**: num. jobs chosen optimally.

- **Serialized time**: time a job would require if ran on a single core.

- **PROOF uses the same resources in a more efficient way**
  - **Grid**: must wait for the last job to finish before collecting results.
  - **PROOF**: workload assignment is independent from the n. of jobs.

- **Speedup**: 28% on a 12 hours job, 18% on a 10 days job.

- **Analytical results**: they represent the upper speedup limit.
The Virtual Analysis Facility

What is the VAF?

- A cluster of CernVM virtual machines: one head node, many workers
- Running the HTCondor job scheduler
- Capable of growing and shrinking based on the usage with elastiq

- Configured via a web interface: cernvm-online.cern.ch
- Entire cluster launched with a single command
- User interacts only by submitting jobs
- Elastic Cluster as a Service: elasticity is embedded, no external tools
- PoD and dynamic workers: run PROOF on top of it as a special case
The VAF is cloud-aware

VAF leverages the CernVM ecosystem and HTCondor

- **CernVM-FS**: all experiments software downloaded on demand, featuring aggressive caching with HTTP proxies
- **CernVM 3**: tiny (< 20 MB) virtual machine image, immediate to deploy: OS on demand with root filesystem from CernVM-FS
- **CernVM Online**: web interface for VM and cluster creation, and secure repository of created configurations
- **HTCondor**: batch system with great scaling capabilities, with worker nodes self-registering to the head node
elastiq is a Python app monitoring the queue to make it elastic

- Jobs waiting too long will trigger a scale up
- Supports minimum and maximum quota of VMs
- You deploy only the master node: minimum quota immediately launches VMs automatically

Integrated in CernVM 3

source: github.com/dberzano/elastiq
A cluster in four steps

• Configure the **head node**
• Configure the **worker nodes**
• Put them together in a **cluster**
• Deploy the **cluster**

Stable and in production: cernvm-online.cern.ch

...then **copy and paste** the generated command

Just click on the **deploy** button...
μCernVM+PROOF startup latency

Measured the delay before requested resources become available

Target clouds:
- Medium: OpenNebula @ INFN Torino
- Large: OpenStack @ CERN (Agile)

Test conditions:
- μCernVM use a HTTP caching proxy
  → Precaching via a dummy boot
- μCernVM image is < 20 MB
  → Image transfer time negligible
- VMs deployed when resources are available
  → Rule out delay and errors due to lack of resources

What contributes to latency
- μCernVM configuration occurring at boot
- First-time software download from CernVM-FS
- HTCondor nodes registration
- PROOF+PoD reaction time
μCernVM+PROOF startup latency

Measured time elapsed between PoD workers’ request and availability: `pod-info -l`

Results are the average of 10 VMs successfully deployed

CERN OpenStack

Torino OpenNebula

Time to wait for workers [m:ss]

<table>
<thead>
<tr>
<th>CERN OpenStack</th>
<th>Torino OpenNebula</th>
</tr>
</thead>
<tbody>
<tr>
<td>3:57</td>
<td>4:36</td>
</tr>
</tbody>
</table>

Maximum 4:36 minutes latency: perfectly acceptable!
Data access for Torino’s VAF

- Data replicated on a dedicated GlusterFS storage
- Current size: 51 TB, 99% full
- Comparable speed using the Grid storage instead of dedicated: more sustainable

Up to cumulative 600 MB/s serving 84 parallel workers

Limit: workers network (1 GbE)
Using the Virtual Analysis Facility

The elastic, on-demand Virtual Analysis Facility is multipurpose

- Runs every task that can be submitted to a queue (not only PROOF)
- Runs on every cloud and respects the environment: unused resources are properly disposed

Usage examples:
- Medical imaging
- All types of Monte Carlos
- Quality Assurance jobs
- Neuroscience (e.g. human brain simulations)

Where can I run it?
- Any private cloud e.g. Torino’s cloud or CERN Agile
- Commercial clouds you pay for e.g. Amazon Elastic Cloud

It can be used now: VAF is production grade, cloud resources are widespread
Outcomes
Private cloud at INFN Torino

- Complete **redefinition** of the **provisioning model**: from Grid to cloud
- Progressive **migration** of Grid nodes and services: now all **new computing resources are “cloudified”**
- **Small groups** can finally get computing power without even accessing the computing center and without leaving resources unused
- Pioneered the setup of a **production-grade private cloud using industry standard tools at INFN**
The ALICE Experiment

ALICE Analysis Facilities: “static” PROOF deployments

• Contributions to the definition of the data access model

• Development of a dataset stager daemon, now part of ROOT, in production since 2010

• The VAF prototype in Torino was the first ALICE PROOF deployment on virtual machines

Towards our Virtual Analysis Facility

• VAF runs on HLT farm to process QA batch jobs (no PROOF involved)

• Forthcoming PROOF deployments will start using VAF
PROOF and CernVM

The VAF involved conjoint work in PROOF and CernVM

PROOF

• Development of the long-awaited **dynamic workers** feature

• Making **PROOF on Demand** work on particular clouds

CernVM

• Development of the support for **multiple contextualization sources**

• Consolidation of **HTCondor** support

• Contribution and maintenance of **CernVM Online**
A High-Energy Physics use case

• Searching \( \Lambda \) hypernuclei (hypertriton and anti-hypertriton) in heavy ion collisions through invariant mass analysis

• Rare events: requires large amount of data to analyze
  \[ \rightarrow \text{storage testing: } \sim 20 \text{ TB for a Pb-Pb LHC 2010 run period} \]

• Many cuts plus topological cuts to improve signal (also TPC+ITS refit!)
  \[ \rightarrow \text{high CPU usage} \]

• Particle identification uses OADB access
  \[ \rightarrow \text{stress-test for network} \]

• Light output data: \sim 80 histograms

• Analysis with S.Bufalino, E.Botta

• Real analysis entirely done on the VAF with PROOF
• Measured the reliability of our cloud infrastructure
• Reference analysis used while developing the VAF
M5L-CAD automatic lung CT analysis

**Fight lung cancer by massive pre-screening and cloud analysis**

- Identification algorithms: RGVP, CAM, VBNA
  → Results and probabilities are then combined

- **SaaS Interface: WIDEN** (Web-based image and diagnosis exchange network) manages the workflow
  → Only a browser required for the Physician

- Positives are notified via email and SMS
  → Number of false positives is low

- First prototype of embedded elasticity
  → It became the main component of the VAF

- Work in collaboration with P. Cerello
A national research project: PRIN

- Computing project involving **11 universities and INFN LNL**, and different LHC experiments
- The proposal has been accepted and it is being funded by the **Department of Education (MIUR)** effective January 2013
- One of the aspects is the **adoption** of the Virtual Analysis Facility as a sustainable analysis model
- This is **pushing** other computing centers to configure their own cloud
- **Torino** is currently providing a **fundamental support** to other computing centers, thanks to the **expertise in cloud computing** acquired during the last three years
Thank you!
Additional slides
Demo: elastic VAF with PROOF in Torino
Demo: elastic VAF with PROOF in Torino