Dynamic provisioning of local and remote compute resources with OpenStack

CHEP 2015 - Okinawa, Japan
Manuel Giffels, Thomas Hauth, Frank Polgart, Günter Quast | 14. April 2015
High Performance Computing (HPC) focuses on the efficient execution of compute intensive, tightly-coupled tasks.*

High Throughput Computing (HTC) focuses on the efficient execution of a large number of loosely-coupled tasks.*

Nearly all High Energy Physics workloads belong to the HTC category

| Property         | typical HPC jobs                        | typical HEP jobs          |
|------------------|-----------------------------------------|----------------------------|
| Interdependence  | interconnect between nodes             | each node independent     |
| Runtime          | runs up to weeks                        | individual job runtime ∼24h|

HEP jobs on HPC Clusters

- HEP batch jobs can be placed anywhere (no fast interconnect between jobs)
- Backfill of HEP jobs can be run also in smaller quantities to fully load a partially occupied cluster

* according to the European Grid Infrastructure (EGI) https://wiki.egi.eu/wiki/Glossary_V1
The Infrastructure-as-a-Service (IaaS) model

- Infrastructure (e.g. machines, network) is virtualized
- Decouples complexities of hardware maintenance and specific software setup
- The life cycle of this virtual infrastructure is managed by a Cloud system:
  - Virtual machine images are managed
  - The user can upload and start custom virtual machines
  - Storage blocks can be attached to these VMs
Our Technology Choices

Goals: mature, off-the-shelf, experiment-independent

Cloud Operating System: OpenStack [1]
- Complete open source IaaS framework
- Backed by lots of big companies (IBM, Red Hat, Rackspace, ...)
- Standardized API (Amazon EC2, fits with HEP workflow tools)
- Established with HPC Centers and experience among admins
- Active Involvement of the HEP Community (CERN Personnel in OpenStack Foundation Board)

Batch Server: HTCondor [2]
- Excels at integration of dynamic resources
- Integrate worker nodes beyond network zone boundaries
- Resilient, scales to more than 10k jobs, open source
  proven software and long term support expected
Enabling Shared Remote HPC Resources for HEP

- German funding agencies encourage sharing of computing resources among user groups
- Acquiring funding for dedicated HEP-only installations becomes more difficult

**bwForCluster cluster installation case study**

- Located at Freiburg University ∼150km south of Karlsruhe
- Shared by 3 diverse user groups: Particle Physics, Neuroscience, Microsystems technology
- State funding secured, full installation with ∼8000 cores is expected to be available in Fall 2015
- Virtualization is a key technology to allow for a efficient sharing among the user groups
- A 10Gbit dedicated link allows for an efficient data transfer (possible to upgrade to 100Gbit) to GridKa Tier-1 site
- Current prototype installation: system similar to the final bwForCluster installation
- Fully functional OpenStack setup with 800 cores
Topology of Local and Remote Sites

KA – Campus North

KA – Campus South

Remote Site

GridKa Storage

Batch Server

HTCondor

Compute Node

Controller Node

- nova-api
- nova-scheduler
- nova-conductor
- keystone-all
- neutron-server
- neutron-ovs-plugin
- neutron-l3-agent
- neutron-dhcp-agent
- glance-api
- glance-registry
- horizon
- mysql-server
- rabbitmq-server
- openvswitch
- apache-server

Manuel Giffels, Thomas Hauth, Frank Polgart, Günter Quast – 14. April 2015
Custom Virtual Machine Image

- VM images are built with the OZ toolkit [3]
  - Installs an OS, software and configurations according to template files (.tdl)
- Advantages:
  - Reproducibility
  - ./buildImage.sh <template file>
  - Easy to adapt working templates to new sites
- We used the CERN OpenStack SL6.5 image templates [4] as starting point:
  https://github.com/cernops/openstack-image-tools

Applied a lot of modifications:
- Added HTCondor which integrates the VM at boot-time into pool
- CVMFS for Grid UI and CMS software
- Disabled yum auto-update (urgent security-fixes are deployed via image rebuild) Created cleaner and smaller image:
- Removal of unnecessary files (/usr/share/man, /usr/share/doc, desktop backgrounds, cups)
- Filled the free disk space with zeros (allows for better compression of the image)
- Final Image size (1.3 Gb, before 2.1 Gb)
One important goal of this system is to allow the Institute’s user group seamless and hassle-free access to both the Institute’s local and the remote Cloud resources.

- Users can submit their jobs using the `condor_submit` command
- The EKP institute also operates a local OpenStack setup, which is utilizing spare resources of desktop machines
- The recommended workflow is to use the job submission tool `grid_control` [5]
- By requiring specific HTCondor ClassAds when submitting the job, users can either:
  - submit only to local Cloud worker-nodes (with direct files server access)
  - submit only to remote Cloud worker-nodes (with file access via Grid tools)
  - submit to both at the same time
Important Questions to answer

Once the remote HPC resources are provisioned via OpenStack and workloads can be send via HTCondor, some important questions need to be answered:

- Stability of the OpenStack hypervisor and virtual machines
- Stability of the HTCondor integration
- Data-transfer rates and stability of the system

The following slides will show some of the studies which have been done to answer these questions for our setup.
Data-transfer from and to Remote Site

- 400 concurrent VMs running at remote site to simulate typical work load
- **Stage-In**: Jobs copy files with the size 250 MB to 1 GB from the GridKa Tier-1 storage element
- **Stage-Out**: Synthetic jobs create random 1 GB files and transfer them to the GridKa Tier-1 storage element

**Stage In**

- Peak Rate: 611.03 MB/s
- Average: 477.13 MB/s

**Stage Out**

- Peak Rate: 955.46 MB/s
- Average: 755.62 MB/s

- Benchmarks the full chain: Hardware virtualization, Firewalls, Routing and storage pools
- **Successful outcome**: The available bandwidth (10 GBit) can be saturated
Long-term example: QCD-Theory computations

- As an example: long-term QCD production
- Concept: use OpenStack VMs to compute and Grid-SE for storage
- HPC / MC workload (QCD interpolation tables) with FastNLO [6]
- High CPU and low IO

more than 100kh of cputime (with FastNLO workflow)
Long-term example: QCD-Theory computations

- The dCache-based Grid storage element at the DESY computing center was used for storage.
- More than 1 TB of input and output data transferred from/to the VMs.
- **Successful outcome**: Processed > 20000 jobs over two month without problems.

Other workloads have also been tested successfully on the local and remote resources:
- Monte-Carlo production with the CMS Software Framework
- ROOT n-tuple based data-analysis
Conclusion

- Virtualization is essential to allow access for the HEP community to shared HPC resources
- Mature software for virtualization (KVM, OpenStack) and flexible work management (HTCondor) exists today
- The bwForCluster project uses virtualization to solve the diverse requirements of its tenants
- First evaluations of the bwForCluster test system successful
- We are convinced the shared cluster model can work on a technical level and bring benefits to our HEP groups

Outlook

- Accompany the bwForCluster procurement with our virtualization requirements in mind
- Discussions on governance and operational details with the other user groups of the bwForCluster are ongoing
References I

“OpenStack website.”
https://www.openstack.org (13.4.2015).

“HTCondor website.”
http://research.cs.wisc.edu/htcondor (13.4.2015).

“OZ website.”
https://github.com/clalancette/oz (13.4.2015).

“CERN OZ templates website.”
https://github.com/cernops/openstack-image-tools (13.4.2015).

“Grid-control website.”
https://ekptrac.physik.uni-karlsruhe.de/trac/grid-control (13.4.2015).

“FastNLO website.”
http://fastnlo.hepforge.org (13.4.2015).
Additional Material
Comparison of VM and bare metal performance

fastNLO benchmark

exec. time ratio

0 1 2 3 4 5
#processes
4 cores EKP VM
4 cores baremetal
4 cores baremetal pinned