

On-demand provisioning of HEP compute resources on cloud sites and shared HPC centers

CHEP 2016 - San Francisco, United States of America

Günther Erli, Frank Fischer, Georg Fleig, Manuel Giffels, Thomas Hauth, Günter Quast, Matthias Schnepf
Institute of Experimental Nuclear Physics (IEKP), KIT, Germany

Jörg Heese (joerg.heese@1und1.de), Katja Leppert, Javier Arnáez de Pedro, Rainer Sträter
1&1 Internet SE, Montabaur, Germany

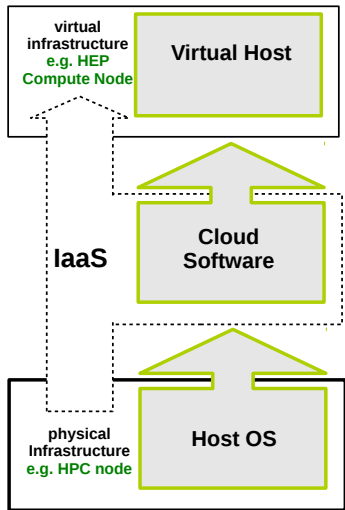


- Modern high energy physics (HEP) research relies heavily on large computing resources:
 - Simulation** CPU intensive, moderate I/O
 - Reconstruction** I/O and CPU intensive
 - Analysis** I/O intensive, moderate CPU usage
 - Other Compute-Intense Jobs like Limit Calculation, Theory Calculations** mostly high CPU, low I/O
- Continuously growing demand for computing resources requires rethinking of traditional HEP-only clusters

New ways to acquire computing capacity

- **HPC Systems**
High Performance Computing (HPC) resources via bare-metal or virtualization
- **Local Opportunistic Resources**
via bare metal, lightweight Docker or virtualization abstraction
- **Commercial Clouds**
Pay commercial providers to handle peak loads or dedicated computation campaigns

From HPC Cluster to HEP Worker Node



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
- HEP software is loaded on-demand via the CVMFS file system
- Input and output data is loaded via fast WAN links from HEP-specific storage sites with SRM or XRootD

Goals: off-the-shelf (if possible), experiment-independent



Batch Server: HTCondor [1]

- Excels at handling dynamic resources
- Can easily integrate worker nodes beyond network zone boundaries
- ClassAd system and custom job routing allows adaption to specific use cases
- Resilient, scales to more than 100k jobs, open source



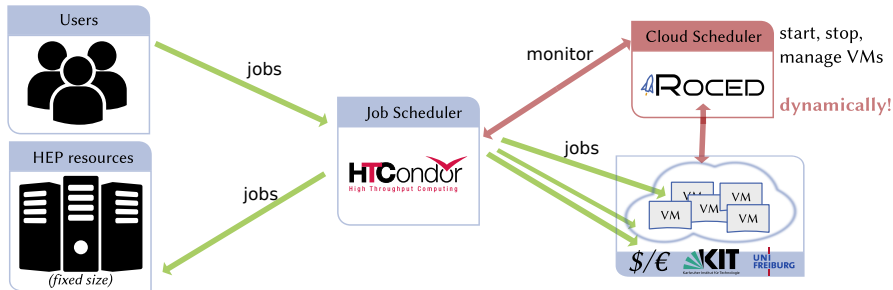
Cloud Scheduler: ROCED [2]

- Support for multiple Cloud APIs (OpenStack, Amazon EC2 and other commercial providers) and batch systems
- Easily extendable thanks to modular design
- Parses HTCondor's ClassAds and boots VMs on cloud sites

For System Abstraction: **OpenStack** and **Docker**

Job Submission and Workflow Management

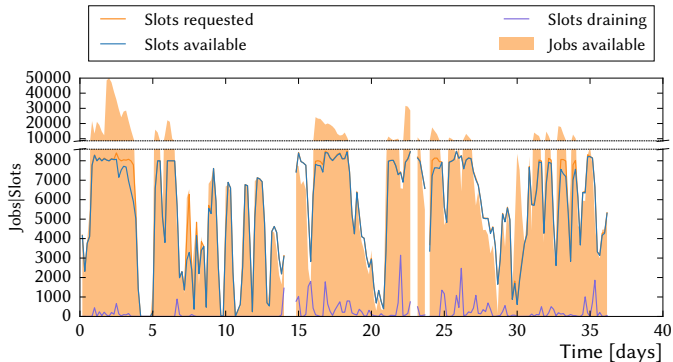
Goal: allow the Institute's user groups seamless and hassle-free access to both the Institute's local and the remote Cloud resources.



- Users can submit their jobs using the `condor_submit` command
- The recommended workflow is to use the job submission tool `grid_control` [3]
- By adding specific HTCondor ClassAds when submitting the job, users can either:
 - submit only to local worker-nodes (with direct fileserver access)
 - submit only to remote Cloud worker-nodes (with file access via Grid tools)
 - submit to both at the same time

- Located at Freiburg University 150km south of Karlsruhe
- Shared by 3 diverse scientific user groups:
Elementary Particle Physics, Neuroscience, Microsystem Engineering
- Full system recently installed with **16,000 Broadwell CPU cores**
- OpenStack used as virtualization manager
- VM scheduling integrated into existing batch system to honor other user groups on cluster → **Hybrid HPC Cluster** [4]
- Special ROCED-Adapter was developed to provision VMs via the HPC Cluster's batch system
- **Important feature**
User jobs are automatically executed on this cluster, if local file system access was not explicitly requested by the users

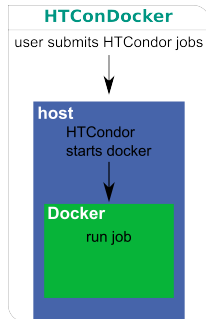
HPC System: Results



- Pre-production cluster (1000 cores) used for 6 month
- Some adaptations were required: more OpenStack management nodes, configuration of HTCondor collector (file handle limit and friends)
- Production system has been scaled up to **11000 virtualized cores**
- In total, more than **5 million CPU hours** of user jobs so far processed

Local Opportunistic: HTCondor+Docker

- New feature in HTCondor: run jobs in a Docker container, we call it **HTConDocker**
- Users submit job via HTCondor
- HTCondor runs on bare-metal system and starts a Docker container on demand
- More lightweight (esp. memory) compared to full virtualization
- Easier administration than OpenStack: only HTCondor and Docker
- Users may select a Docker container best suited for their needs



Positive Experience so far

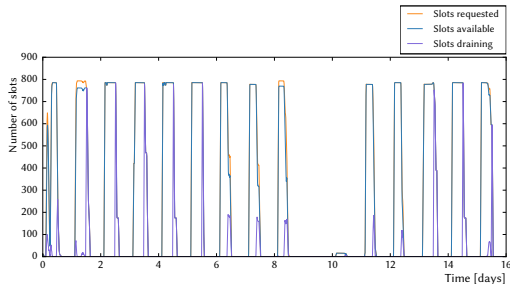
- Used successfully on our Institute's powerful Desktops (4 core, 16 GB RAM) to provide an additional 150 opportunistic job slots
- More than **10k CPU hours of computing over one weekend**
- Satisfies the requirements of various user groups: AMS, Belle II, CMS

- 1&1 Internet SE is one of Europe's leading internet service provider with a strong global presence
- Jointly 1&1 and the KIT team started a pilot project to evaluate the Cloud Server product for HEP jobs
- The Cloud Server product offers dynamic provisioning of VMs and accurate billing depending on the machines uptime and configuration
- With the goal to make the results available to the scientific community, we adapted the 1&1 standard product to HEP use cases



Scope:

- ROCED-Adapter for Cloud Server API [5] was developed
- Possibility to upload and deploy a custom VM-Image with Scientific Linux 6.7 (or other, if needed) and CVMFS and HTCondor support
- Configured dedicated CVMFS-Squid VM in 1&1 data center which gets automatically started by ROCED as soon as one worker VM is booted
- Possibility to run workloads across 1&1's data centers



Resulting preconditions:

- We took advantage of the fact that the load distribution is lower during night time (due to typical customer profile)
- Only jobs with a run-time smaller than 12h were scheduled to 1&1 data centers

Results:

- Per night up to **800 job slots** were provisioned, if enough user jobs had been queued
- Jobs of users from multiple experiments **run reliably on the virtual WN**
- API based scheduling without manual intervention is possible

Conclusion

- Our institute runs a flexible computing system which is able to leverage resources from multiple sources:

Shared HPC System

Local Opportunistic Resources

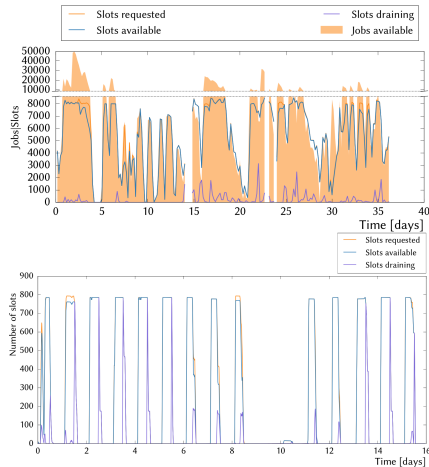
1&1 Internet Cloud Server


- The complexity and provisioning of these (remote) resources is hidden from the user with:


HTCondor - for job submission and management


ROCED - for provisioning of remote resources


- This dynamic model gradually replaces our Institute's private HEP-only cluster in the basement: **both in ease of use and capacity**




 “HTCondor website.”
<http://research.cs.wisc.edu/htcondor> (28.9.2016).

 “ROCED website.”
<https://github.com/roced-scheduler/ROCED> (28.9.2016).

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

 “ACAT 2016: Dynamic provisioning of a HEP computing infrastructure on a shared hybrid HPC system.”
<https://indico.cern.ch/event/397113/contributions/1837774/> (28.9.2016).

 “1&1 Cloud Server SDK Python.”
<https://github.com/1and1/oneandone-cloudserver-sdk-python> (29.9.2016).

Additional Material

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

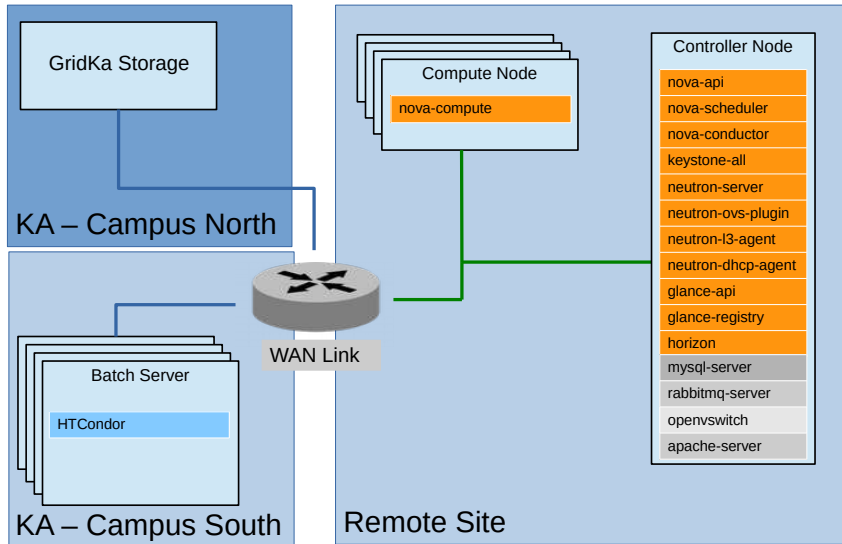
Property	typical HPC jobs	typical HEP jobs
Interdependence	interconnect between nodes	each node independent
Runtime	runs up to weeks	individual job runtime ~24h

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

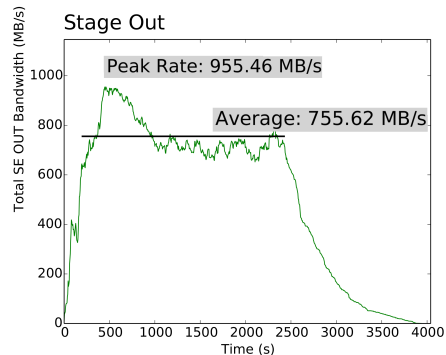
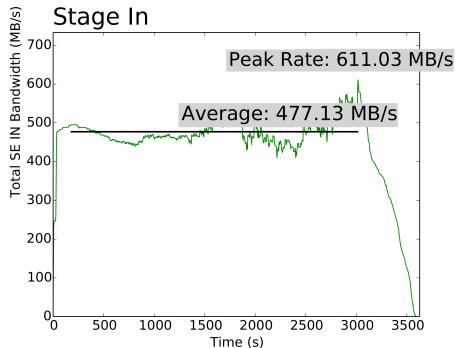
* according to the European Grid Infrastructure (EGI) https://wiki.egi.eu/wiki/Glossary_V1

Topology of Local and Remote Sites



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



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