Contribution ID: **472** Type: **Oral**

Geographically distributed Batch System as a Service: the INDIGO-DataCloud approach exploiting HTCondor

Wednesday 12 October 2016 12:00 (15 minutes)

One of the challenges a scientific computing center has to face is to keep delivering a computational framework well consolidated within the community (i.e. the batch farm), while complying to modern computing paradigms. The aim is to ease system administration at all levels (from hardware to applications) and to provide a smooth end-user experience.
HTCondor is a LRMS widely used in the scientific community and it's Cloud aware (i.e. it does not resent from a volatile environment where resources might dynamically change). Apache Mesos is a tool that allows to abstract computing resources away from the physical or virtual hosts and to deal with the entire computing infrastructure as a single pool of resources to be shared among services.
Within the INDIGO-DataCloud project, we adopted two different approaches to implement a PaaS-level, on-demand Batch Farm Service based on HTCondor and Mesos.
In the first approach, the various HTCondor daemons are packaged inside pre-configured Docker images and deployed as Long Running Service (LRS) through Marathon, profiting from its health checks and failover capabilities.
In the second approach, we have implemented an HTCondor framework for Mesos, that can be used by itself or as a component in the more complex INDIGO PaaS system in conjunction with an orchestration layer like i.e. Marathon. The new framework consists of a scheduler to implement HTCondor policies on the resource offers provided by the Mesos master and a dedicated executor to launch tasks on the slave nodes. The benefits of an ad-hoc framework are first of all a fine-grained level of control on the tasks the application is responsible for. Moreover, it is possible to implement the preferred authorization rules and roles for multi-tenancy and to define application-specific scaling rules. Application isolation and packetization are achieved with the Docker Containerizer module of Mesos.
The most difficult aspects of both approaches concern networking and storage. For instance, the usage of the shared port mode within HTCondor has been evaluated in order to avoid dynamically assigned ports; container-to-container communication and isolation have been addressed exploring solutions based on overlay networks (including e.g. the Calico Project implementation).
Finally, we have studied the possibility to deploy an HTCondor cluster that spans over different sites, also exploiting the Condor Connection Brokering (CCB) component that allows communication across a private network boundary or firewall, as in case of multi-site deployments.
Concerning the storage aspects, where factors such as scalability, performance and reliability have to be taken into account, we have explored the usage of CVMFS (using Parrot) and the integration with the INDIGO Data Services (Onedata, Dynafed, FTS).
In this contribution, we are going to describe and motivate our implementative choices and to show the results of the first tests performed.

Tertiary Keyword (Optional)

Network systems and solutions

Secondary Keyword (Optional)

Computing middleware

Primary Keyword (Mandatory)

Distributed workload management

Primary authors: COSTANTINI, Alessandro (University of Perugia); ITALIANO, Alessandro Italiano (INFN-CNAF); MICHELOTTO, DIEGO (INFN - National Institute for Nuclear Physics); SALOMONI, Davide (Universita e INFN, Bologna (IT)); AIFTIMIEI, Doina Cristina (INFN - CNAF, IFIN-HH); DONVITO, Giacinto (Universita e INFN, Bari (IT)); GAIDO, Luciano (Universita e INFN Torino (IT)); ANTONACCI, Marica; PANELLA, Matteo (CNAF); CABALLER, Miguel (UPV); VALLERO, Sara (Universita e INFN Torino (IT)); BAGNASCO, Stefano (Universita e INFN Torino (IT)); BOCCALI, Tommaso (Universita di Pisa & INFN (IT))

Session Classification: Track 3: Distributed Computing

Track Classification: Track 3: Distributed Computing