20th International Conference on Computing in High Energy and Nuclear Physics (CHEP2013)

Contribution ID: **164** Type: **Oral presentation to parallel session**

Dynamic VM provisioning for Torque in a cloud environment

Tuesday, October 15, 2013 3:45 PM (22 minutes)

The Nectar national research cloud provides compute resources to Australian researchers using OpenStack. CoEPP, a WLCG Tier2 member, wants to use Nectar's cloud resources for Tier 2 and Tier 3 processing for ATLAS and other experiments including Belle, as well as theoretical computation. CoEPP would prefer to use the Torque job management system in the cloud because they have extensive experience in managing Torque and users are familiar with running batch jobs using Torque. However, Torque was developed for static clusters, and worker nodes cannot easily be dynamically added or removed, and this also requires updating related services such as monitoring servers like Nagios and Ganglia, file service and package management service.

A number of projects have been looking for an easy way to run batch jobs in the cloud. The solution described here is inspired by two successful projects: ViBatch, which enables Torque to use a local KVM hypervisor to add and remove virtualized resources, and Cloud Scheduler, which dynamically allocates cloud VMs according to the workload of an associated Condor queue. This generalised solution combines the advantages of the above projects and enables cloud-based dynamic VM provisioning for Torque. It includes two parts: prologue, epilogue and a number of utility scripts working alongside Torque; and a service called VM Pool that maintains an elastic set of VMs in the cloud.

A special Torque worker node is configured with the aforementioned scripts in order to communicate with VM Pool for job execution. When a new job comes in, prologue script requests a VM from VM Pool, and does some initialization on it, including sending the job description file to the VM, creating relevant credentials on the VM and setting up environment variables. Then Torque executes the job on that VM through an SSH connection. When finished, epilogue returns the VM back to VM Pool, and cleans up all relevant data.

VM Pool runs as a standalone service to orchestrate VMs in the cloud. The number of VMs in the pool changes according to the number of VM requests from the worker node but remains between a specified minimum and maximum value. Any VM in the pool has three states: Free, Unusable and Busy. Unusable VMs can be dead, in building state, in deleting state, or in active state but inaccessible due to network or other issues. Unusable VMs are checked periodically to see if the state is changed, if they remain in this state for longer than a specific time, they will be terminated.

Torque sees a single worker node with a fixed number of processors (the maximum number specified in VMPool), so resources do not need to be dynamically added or removed from Torque, this is handled in VM Pool.

Currently a Beta system has been implemented and tested on the Nectar cloud. Next it will be under User Acceptance Test, then once everything works fine it will become a production system for Tier 3 and Tier 2.

Author: Dr ZHANG, Shunde (eRSA, CoEPP)

Co-authors: SEVIOR, Martin (University of Melbourne (AU)); Dr CODDINGTON, Paul (eRSA)

Presenter: BOLAND, Lucien (University of Melbourne)

Session Classification: Distributed Processing and Data Handling A: Infrastructure, Sites, and Virtualization

Track Classification: Distributed Processing and Data Handling A: Infrastructure, Sites, and Virtualization