Data intensive high energy physics analysis in a distributed cloud

A. Charbonneau¹, A. Agarwal², M. Anderson², P. Armstrong², K. Fransham², I. Gable², D. Harris², R. Impey¹, C. Leavett-Brown², M. Paterson², W. Podaima¹, R.J. Sobie² and M. Vliet²

¹ National Research Council Canada
100 Sussex Drive, Ottawa, Canada
² Department of Physics and Astronomy
University of Victoria, Victoria, Canada V8W 3P6
E-mail: Andre.Charbonneau@nrc-cnrc.gc.ca

Abstract. We show that distributed Infrastructure-as-a-Service (IaaS) compute clouds can be effectively used for the analysis of high energy physics data. We have designed a distributed cloud system that works with any application using large input data sets requiring a high throughput computing environment. The system uses IaaS-enabled science and commercial clusters in Canada and the United States. We describe the process in which a user prepares an analysis virtual machine (VM) and submits batch jobs to a central scheduler. The system boots the user-specific VM on one of the IaaS clouds, runs the jobs and returns the output to the user. The user application accesses a central database for calibration data during the execution of the application. Similarly, the data is located in a central location and streamed by the running application. The system can easily run one hundred simultaneous jobs in an efficient manner and should scale to many hundreds and possibly thousands of user jobs.

1. Introduction

Infrastructure as a Service (IaaS) cloud computing is emerging as a new and efficient way to provide computing to the research community. The growing interest in clouds can be attributed, in part, to the ease of encapsulating complex research applications in Virtual Machines (VMs) with little or no performance degradation [1]. Studies have shown, for example, that high energy physics application code runs equally well in a VM or on the native system [2]. Virtualization technologies not only offers several advantages such as abstraction from the underlying hardware and simplified application deployment, but in some situations where traditional computing clusters have hardware and software configurations which are incompatible with the scientific application’s requirements, virtualization is the only option available. A key question is how to manage large data sets in a cloud or distributed cloud environment.

We have developed a system for running high throughput batch processing applications using any number of IaaS clouds [3]. This system uses a number of off-the-shelf software such as Condor [4] and Nimbus [5], in addition to custom components such as a cloud scheduling element and a VM image repository. The results presented in this work use the IaaS clouds at the National Research Council (NRC) in Ottawa and two clouds (elephant and hermes) at the University of Victoria (UVIC). The Amazon EC2 cloud was not included in these tests because of network
performance issues between the NRC and UVIC clouds and Amazon. The total amount of memory and CPU of each computational cluster in our clouds are divided evenly into what we call VM slots, where each of these slots can be assigned to run a VM. When a VM has finished running, that slot’s resources are then released and available to run another VM. In other words, the number of VM slots in our system determines the maximum number of simultaneously running VMs at any given time. A total of 84 VM slots on UVIC and NRC clouds were used in generating the results presented later in this paper. Two additional clouds have since been added to the system raising the capacity to over 200 VM slots. The input data and analysis software are located on one of the UVIC clouds and the VM images are stored in a repository on the NRC cloud. The Canadian sites are connected by a research network provided by CANARIE while the commodity network is used to connect the NRC and UVIC clouds to Amazon EC2.

We describe the system architecture in the next section, however, here we highlight the main features. Users are provided with a set of VMs that are configured with the application software. The user adds code to run their specific analysis and saves the VM to a repository. The user submits their jobs to a Condor scheduler where the job script contains a link to the required VM. A cloud scheduling component we implemented (called Cloud Scheduler [3]) searches the job queue, identifies the VM required for each queued jobs, and sends out a request to one of the clouds to boot the user specific VM. Once the VM is booted, the scheduler submits the user job to the running VM. The job runs and returns any output to a user specified location. If there are no further jobs requiring that specific VM, then Cloud Scheduler shuts it down.

The system has been demonstrated to work well for applications with modest I/O requirements such as the production of simulated data [6]. The input files for this type of application are small and the rate of production of the output data is modest (though the files can be large). The system has also been used for the analysis of astronomical survey data where the images are pre-staged to the storage on the cloud [6].

In this work, we focus on data intensive high energy physics applications where the job reads large sets of input data at higher rates. In particular, we use the analysis application of the BaBar experiment [7] that recorded electron-positron collisions at the SLAC National Accelerator Laboratory from 2000-2008. We show that the data can be quickly and efficiently streamed from a single data storage location to each of the clouds. We will describe the issues that have arisen and the potential for scaling the system to many hundreds or thousands of simultaneous user jobs.

2. System architecture
The architecture of the distributed IaaS cloud system is shown in fig. 1 (see ref. [3] for a detailed description). The UVIC and NRC clouds use the Nimbus software to manage VMs while Amazon EC2 uses its own proprietary software. The system is designed so that the user submits their jobs to a single Condor scheduler. The system then boots the user-specific VMs on any one of the available clouds.

Users submit jobs using X.509 Proxy Certificates [8] to authenticate. The certificates are also used to authenticate with Nimbus clusters when starting, shutting down, or polling VMs. Authentication with EC2 is done by using a standard shared access key and secret key.

To simplify the management of user VMs we have developed a system called Repoman [9]. Repoman implements an online VM repository where the user can retrieve copies of a variety of standard VM images preloaded with the application software. The user clones the standard VM, adds and tests their software, saves it in the VM repository and includes a reference to their VM image in their job submission script.

The user submits their jobs to the Condor job scheduler. We selected Condor as the job schedule since it was designed to utilize heterogeneous idle workstations and is an ideal job scheduler for a dynamic VM environment where worker nodes become available on demand.
Figure 1. An overview of the architecture used for the system. A user prepares a VM image and a job script. The job script is submitted to the job scheduler. The Cloud Scheduler reads the job queue and makes a request to boot the user VM on one of the available clouds. Once there are no more user jobs requiring that VM type, the Cloud Scheduler makes a request to the proper cloud to shutdown the user VM.

Users submit jobs by issuing the *condor_submit* command. The user must add a number of additional parameters specifying the location of the image and the properties of the required VM.

The management of the VMs on the clouds is done by Cloud Scheduler [3]. Cloud Scheduler monitors the queue of jobs and if one of the jobs needs a VM that is not already booted on one of the clouds, then it sends a request to the next available cloud. Cloud Scheduler will also shut down a VM if there are no jobs needing that particular type of VM anymore. A complete description of Cloud Scheduler can be found in ref. [3].

3. Database and data management

Analysis jobs in high energy physics typically require two inputs: event data and configuration data. The configuration data also includes a BaBar conditions database [10], which contains time-dependant information about the conditions under which the events were taken. The event data can be the real data recorded by the detector or simulated data. Each event contains information about the particles seen in detector such as their trajectories and energies. The real and simulated data are nearly identical in format; the simulated data contains additional information describing how it was generated. The user analysis code analyzes one event at a time. In the BaBar experiment the total size of the real and simulated data is approximately 2
PB but users typically read a small fraction of this sample. In this work we use a subset of the data containing approximately 8 TB of simulated and real data. The configuration data that describes the state of the detector and the conditions database totals 24 GB.

The event data for this analysis was stored in a distributed Lustre file system at UVIC. The Lustre file system is hosted on a cluster of six nodes, consisting of a Management/Metadata server (MGS/MDS), and five Object Storage servers (OSS). Each node has dual quad core Intel Nehalem processors and 24 GB of memory. All servers have six disk drives: two 450 GB, 15K RPM, Serial-Attached SCSI (SAS) system drives in a RAID 1 configuration; and four metadata/object storage drives in a RAID 5 configuration. The drives for the metadata storage on the MGS/MDS are 450 GB, 15K RPM, SAS drives. The drives for the object storage on each OSS are 1 TB, 7200 RPM, Serial ATA (SATA) drives. Originally configured to allow remote access via mounting over the public network, the Lustre file system uses a single gigabit interface/VLAN to communicate both internally and externally. This is an important consideration for the test results presented, because these same nodes also host the IaaS front-end (MGS/MDT server) and Virtual Machine Monitors (OSS servers) for the UVIC cloud.

The jobs use Xrootd [11] to read the data. Xrootd is a file server providing byte level access and is used by many high energy physics experiments. Xrootd provides read only access to the Lustre data (read/write access is also possible) and the capability of implementing Grid Security Infrastructure (GSI) authentication. Though the implementation of Xrootd is fairly trivial, some optimization was necessary to achieve good performance across the network: a read-ahead value of 1 MB and a read-ahead cache size of 10 MB was set on each Xrootd client.

The VM images are stored at NRC in a Repoman server and propagated to the worker nodes by http. For analysis runs that includes the Amazon EC2 cloud, we store another copy of the VM images on Amazon EC2 in an Amazon S3 bucket using the ec2_bundle_vol and ec2_upload_bundle tools provided by the Amazon EC2 API Tools package.

In addition to transferring the input data on demand using Xrootd, the BaBar software is also staged to the VMs on demand using a specialized network file system called CernVM File System (CernVM-FS) [12]. Using CernVM-FS can considerably reduce the size of the VM images since the software files are stored in a remote location and read by the application at runtime. The rationale of using CernVM-FS for this project was to reduce the amount of data initially transferred to the clouds when the VM starts by reducing the size of the VM images transferred from the image repository to each cloud site. This not only makes the VM start faster, but also helps mitigate the network saturation after job submission by postponing some of the data transfer to happen later after the job has started.

4. Results
A Xen VM image based on Scientific Linux SL release 5.5 (Boron) was created and the latest release of BaBar software was installed. The BaBar software, which contains C++ and FORTRAN code, requires 1 GB of RAM and is completely contained in a 16 GB VM. Using CernVM-FS to load the BaBar software on demand, we were able to bring down the size of the VM to approximately 4GB. In this study, a member of the team and also a researcher in the BaBar Collaboration added one of their analysis codes to this VM. After compiling, linking and testing the code, the modified VM was saved for access by the batch system. A server was setup as a login node which contained the tools to allow users to submit and manage their jobs. A separate VM contained both the Condor job scheduler and Cloud Scheduler for the tests presented in this paper.

A typical user job in high energy physics reads one event at a time where the event contains the information of a single particle collision. Electrons and positrons circulate in opposite directions in a storage ring and are made to collide millions of times per second in the centre of the BaBar detector. The BaBar detector is a cylindrical detector with a size of approximately 5 meters in
Figure 2. The number of running, queued and failed jobs (a), the network traffic for the transfer of VM images from NRC (b), the traffic for the loading of the required BaBar software files at runtime using CernVM-FS (c), and the traffic from the Lustre file system at UVIC which is used to store the event data (d).
each dimension. The detector measures the trajectories of charged particles and the energy of both neutral and charged particles. A fraction of those events are considering interesting from a scientific standpoint and the information in the detector is written to a storage medium. The size of the events in BaBar are a few kilobytes depending on the number of particles produced in the collision.

The user code analyzes each event independently and writes out a subset of the information into an output file for subsequent analysis. A typical job reads millions of events (the size of each event is approximately 3 KB) and the user submits tens to hundreds of these jobs in parallel. The output files are a few percent of the original input data set size, making it easier to study the events of particular interest to the researcher.

In this work, the analysis is focused on a particular subset of the BaBar data set where there are a small number of particles produced in each collision. We use two separate samples for our tests which we label Tau11-data and Tau11-MC. The Tau11-data and Tau11-MC samples are identical except that the first is real data and the second is simulated data.

We submitted approximately 170 analysis jobs (80 jobs analyzed the real data sample and 90 analyzed the simulation data sample) where a single job runs for approximately 12 hours. The input rate ranges up to a few megabits/second depending on whether the jobs are running on a site local to the data repository or at a remote location. In addition, the processing time is different for each sample as the fraction of interesting events in each sample is different. For example, background or non-interesting events are quickly identified and discarded so that the job does not waste processing time.

Plot (a) of fig. 2 shows the number of VMs on the system over a 3 day period. The yellow curve is the number of running VMs, the blue curve is the number of queued jobs and the orange curve is the number of jobs in an error state. There were 44 VM slots at NRC Ottawa and 40 VM slots at the two University of Victoria clusters. Initially, only the 2 clusters at UVIC were online; the NRC cluster was offline for maintenance. This explains why the number of running VMs plateaus at 40 until approximately 22:00 on Thursday, at which point the NRC cluster came online, adding 44 extra resources to the cloud. The Cloud Scheduler then started to submit jobs to the NRC resources and at approximately 3:00 on Friday all the 84 slots were allocated and running jobs. The 40 VMs required at UVIC corresponds to nearly 160 GB and took approximately 1 hour to transfer from NRC (3000 km away). We are working on a method so that only a single copy of a VM needs to be transferred. It is interesting to note that without the use of CernVM-FS, this data transfer would have been in the range of 640 GB. At around 7:00 on Saturday, we can see that there are approximately 20 jobs in error state. Further investigation revealed that these jobs failed because of a communication failure between the Condor processes, most probably caused by an overly congested network (we are investigating solutions to this problem after discussions with Condor developers). The system kept the failed jobs in the queue and these jobs were eventually resubmitted to the cloud at a later time and ran to completion.

Plot (b) of fig. 2 shows the network traffic at the NRC cloud generated by the transfer of VMs from the Repoman image repository to the clouds. The 40 VMs required at UVIC were obtained from NRC and the blue curve shows the outbound network traffic of approximately 500 Mb/s over 1 hour period (the NRC site has a 1 Gb/s connection to the CANARIE research network in Canada). Once all the VMs were booted, the outbound network traffic at NRC fell to zero until the NRC cluster came online late Thursday, at which point 44 additional VMs were transferred to the available NRC resources. Since this later transfer was done internally on the NRC network, it peaked at a much higher rate of approximately 1.4 Gb/s.

In plot (c) of fig. 2 we observe that the network traffic generated by the VM retrieving the required software files at runtime using CernVM-FS. This figure shows a high throughput transfer which peaked at 900 Mb/s for the VMs running at UVIC and, as expected, a lower transfer...
rate for the files sent to the VMs which started later at NRC. Because CernVM-FS implements some client-side caching, we see that most of the files are transferred soon after the VMs start, and then the network traffic goes down considerably, even though these files are still used by the running application.

Plot (d) of fig. 2 shows the network bandwidth for the data transfers from the UVIC cluster that hosts all the BaBar data. The total bandwidth plateaus around 330 Mb/s for the 84 running jobs. We see an unusual amount of input traffic to the Xrootd server (shown by green histogram). This is abnormal because these type of jobs typically read data from the Xrootd server but write the results at another location. After further investigation, it was found that this anomaly was caused by a faulty network configuration, which was later rectified.

Within a given sample, we find that the time to process the data is affected by the location of the data repository. Jobs that ran at UVIC, which had access to the Xrootd server locally on the network, ran 2 to 3 times faster than jobs that ran on the NRC cloud in Ottawa. Since the data rate out of the Xrootd server at UVIC was not the limiting factor (see plot (d) of fig. 2), this suggests that the overall Xrootd performance is somewhat sensitive to network latency. Investigation is underway on how to further improve Xrootd performance when transferring data over long distances.

One of the features of the system is its ability to recover from faults arising either from local system problems at each of the clouds or network issues. We list some of the problems we identified in the processing of the jobs. For example, we find that:

- A number of VMs at NRC went into an error state (see the orange line in plot (a) of fig. 2 at Friday 22:00) and this was attributed to some error with the Condor Connection Brokering (CCB). We believe that this problem was caused by connection timeouts that were set to values too low to accommodate for the high load on the network at that time. In any case, Cloud Scheduler killed those VMs and held these job in the queue for further analysis by the user.
- Cloud resources can be brought down for maintenance and back up again. In our test, the NRC cloud resources were added to the pool of resources after the set of jobs was submitted. The Cloud Scheduler automatically detected the new resources available and successfully scheduled jobs to these newly available resources without affecting already running jobs.

The overall system has performed remarkably well. We are planning to upgrade the network of the data storage system to 10 Gb/s and this would allows us to scale the number of jobs by an order of magnitude. There are also plans in 2011 to connect Amazon EC2 to the Canadian research network (provided by CANARIE).

A significant fraction of the data transfer in this test was the movement of VMs. It is true that as the number of available VM slots increases, the amount of time taken to transfer the VMs for larger sets of jobs also grows and as a consequence can lead to under-utilization of compute resources. This was alleviated by using CernVM-FS to reduce the VM size, but there is still room for further improvement. We will soon implement a VM caching system so that only a single copy of the VM image would need to be transferred to each cloud, significantly reducing the start-up time of the VMs at remote sites.

5. Conclusion
We have presented the results for running a data intensive particle physics applications in a distributed compute cloud. The results show that multi-TB data sets can be easily accessed from remote locations over a high-speed research network. Further improvements can also be achieved by loading software on demand using a specialized network file system such as CernVM-FS, which not only makes the VMs start faster, but also alleviates network saturation by averaging out the overall bandwidth requirement over a longer period. There appears to
be no limitations to scaling the number of jobs by an order of magnitude once we increase the network bandwidth from the event storage servers and implement a VM image caching mechanism. From the users perspective, the system is robust and is able to handle intermittent network issues gracefully. We have shown that the use of distributed compute clouds can be an effective way of analyzing large research data sets.

Acknowledgments
The support of CANARIE, the Natural Sciences and Engineering Research Council, the National Research Council of Canada and Amazon are acknowledged.

References