Adding IaaS Clouds to the ATLAS Computing Grid

Ashok Agarwal, Frank Berghaus, Andre Charbonneau, Mike Chester, Asoka de Silva, Ian Gable, Joanna Huang, Colin Leavett-Brown, Michael Paterson, Randall Sobie, Ryan Taylor
Outline

I. Motivation
II. Building a “Grid of Clouds”
III. Powered by Cloud Scheduler
IV. New Development: Dynamic Squids
V. Summary
I. Motivation

Announcement of boson discovery

World Wide - prod_running - year

Range from Tue Mar 20 12:00:00 2012 UTC to Wed Mar 20 12:00:00 2013 UTC
Generated by TRIUMF-LCG2 (times in UTC)
I. Motivation

“Just-in-time computing”?

Announcement of boson discovery
I. Motivation

1. Allow commercial cloud bursting for urgent deadlines
   - Costs $$$, but on-time discovery announcements are priceless

2. Augment steady-state grid capacity with non-commercial cloud resources
   - Both public and private

3. Enable WLCG sites that wish to convert to cloud
   - e.g. Australia-ATLAS T2 on nectar

   • Scope of this talk:
     • Adding extra cloud resources, not changing existing grid sites
     • MC production jobs only (light I/O)
II. Building a “Grid of Clouds”

Using Condor and Cloud Scheduler
Grid Job Flow

User ➔ Submit ➔ Panda Queue ➔ Run ➔ Worker Node

User Job

Pilot Factory ➔ Submit ➔ Compute Element ➔ Submit ➔ Batch System

Pilot Job
Grid Job Flow

- Compute Element is tightly coupled to batch system
Cloud Job Flow (on the Grid)

- Cloud Scheduler is loosely coupled to cloud interface
Cloud Job Flow (on the Grid)

- Easy to connect and use many clouds
Connecting Additional Clouds

• Just add a few lines to config file
  • `/etc/cloudscheduler/cloud_resources.conf`
    ```
    [MyCloud]
    host: mycloud.example.org
    cloud_type: OpenStack
    vm_slots: 50
    networks: private
    enabled: true
    ```

• Get authorization on the cloud
  • Secret key or x509 proxy

• Test booting VMs

• Done!
Implications

- Cloud Scheduler is a layer above the resources
- Can access arbitrarily many resource sites, using arbitrarily few Cloud Scheduler servers
  - (within practical limits)
- No ATLAS-specific configuration or services needed at resource site
  - Anyone can contribute to ATLAS computing
  - Don't have to become a T2
Pilot Factory vs. Cloud Scheduler
We believe this approach is

- Simpler to set up
- Easier to maintain and operate
- More scalable and flexible
Participating Clouds
Cloud Queues

- **Australia-NECTAR**: commissioned Dec. 2012
- Fully integrated into grid operations, monitoring, etc.

Total of 300k MC jobs completed over past 12 months
III. Powered by Cloud Scheduler

- Cloud Scheduler is a simple python package for managing VMs on IaaS clouds, based on the requirements of Condor jobs
- Users submit Condor jobs, with additional attributes specifying VM properties
- Developed at UVic and NRC since 2009
- Used by BaBar, CANFAR, as well as ATLAS
  - https://github.com/hep-gc/cloud-scheduler
  - http://cloudscheduler.org/
Condor Job Description File

Executable = runpilot3-wrapper.sh
Arguments = -s IAAS -h IAAS-cloudscheduler -p 25443 -w https://pandaserver.cern.ch -j false -k 0

Requirements = VMTYPE =?= "pandacernvm" && Target.Arch == "X86_64"

+VMName = "PandaCern"
+VMLoc = "http://images.heprc.uvic.ca/images/cernvm-batch-node-2.6.0-4-1-x86_64.ext3.gz"
+VMMem = "18000" #MB
+VMCPUcores = "8"
+VMStorage = "160" #GB
+TargetClouds = "FGHotel,Hermes"

x509userproxy = /tmp/atprd.proxy
Research and Commercial clouds made available through a cloud interface.

- Supported cloud types:
  - Nimbus
  - OpenStack
  - StratusLab
  - OpenNebula
  - Amazon EC2
  - Google Compute Engine (new)
Step 2

User submits a Condor job. The scheduler might not have any resources available to it yet.
Step 3

Cloud Scheduler detects waiting jobs in the Condor queue, and makes a request to boot VMs matching the job requirements.
The VMs boot, attach themselves to the Condor queue and begin draining jobs. VMs are kept alive and re-used until no more jobs require that VM type.
Key Features of Cloud Scheduler

• Generic tool, not grid-specific.

• Dynamically manages quantity and type of VMs in response to user demand.

• Easily connects to many IaaS clouds, and aggregates their resources together.

• Complete solution for harnessing IaaS resources in the form of an ordinary Condor batch system.

• `pip install cloud-scheduler`
IV. Dynamic Squids

- ATLAS uses CVMFS to provide software
- CVMFS uses squid proxies for caching
- There should be a squid VM in each cloud used
  - > 1 if scaling massively
Phantom Boots Squids Dynamically

- Define metrics
- Phantom triggers scaling of VMs based on metrics
Shoal Tracks Squids Dynamically

github.com/hep-gc/shoal

New squids discovered

List of Active Squids

3 active in the last 180 seconds

<table>
<thead>
<tr>
<th>#</th>
<th>Public IP</th>
<th>Private IP</th>
<th>Bytes Out</th>
<th>City</th>
<th>Region</th>
<th>Country</th>
<th>Latitude</th>
<th>Longitude</th>
<th>Last Received</th>
<th>Alive</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>149.165.148.123</td>
<td>1 kB/s</td>
<td>Bloomington IN</td>
<td>United States</td>
<td>39.2499</td>
<td>-86.4555</td>
<td>2s</td>
<td>0h1m37s</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>149.165.148.125</td>
<td>0 kB/s</td>
<td>Bloomington IN</td>
<td>United States</td>
<td>39.2499</td>
<td>-86.4555</td>
<td>3s</td>
<td>0h0m35s</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>149.165.148.127</td>
<td>35593 kB/s</td>
<td>Bloomington IN</td>
<td>United States</td>
<td>39.2499</td>
<td>-86.4555</td>
<td>13s</td>
<td>0h39m36s</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Missing squids removed

- A group of squid is called a “shoal”
- Too bad it isn’t a “squad” :(

© University of Victoria || Visit GitHub Project
VMs Find Nearest Squid

- Query Shoal server
- GeoIP used to find nearest squid to requestor
  - i.e. in the same cloud
- CVMFS configured to use that squid
V. Summary

- Developed and deployed a method to run ATLAS grid jobs in IaaS clouds
- Worked with Australian partners to enable cloud jobs for Australia-ATLAS T2 on nectar
- Delivering beyond-pledge resources to ATLAS using many clouds
  - 300k MC simulation jobs over last 12 months
  - More clouds, queues to come in future

rptaylor@uvic.ca
VM Image

- Dual-hypervisor image, can run on KVM or Xen
- Customized CernVM batch node v2.6.0
- Use whole-node VMs for better efficiency
  - cache sharing instead of disk contention
  - fewer image downloads when ramping up
CA production activity since Jan. 1

Completed jobs (Sum: 1,849,262)
TRIUMF-LCG2 - 55.97%

- TRIUMF-LCG2 - 55.97% (1,034,970)
- SFU-LCG2 - 7.80% (144,203)
- CA-SCINET-T2 - 7.06% (130,584)
- IAAS - 3.35% (61,957)
- AUSTRALIA-NECTAR - 1.44% (26,616)
- CA-MCGILL-CLUMEQ-T2 - 8.45% (156,250)
- CA-VICTORIA.WESTGRIDT2 - 7.65% (141,483)
- CA-ALBERTA.WESTGRIDT2 - 5.10% (94,263)
- AUSTRALIA-ATLAS - 3.19% (58,936)
IAAS

- Early tests Oct. 2011, standard operation since April 2012
Implementation Details

- **Condor Job Scheduler**
  - VMs contextualized with Condor Pool URL and service certificate
  - VM image has the Condor startd daemon installed, which advertises to the central manager at start
  - GSI host authentication used when VMs join pools
  - User credentials delegated to VMs after boot by job submission
  - Condor Connection Broker handles private IP clouds

- **Cloud Scheduler**
  - User proxy certs used for authenticating with IaaS service where possible (Nimbus). Otherwise using secret API key (EC2 Style).
  - Can communicate with Condor using SOAP interface (slow at scale) or via condor_q
Credential Transport

- Securely delegates user credentials to VMs, and authenticates VMs joining the Condor pool.