Clusters in the Jetstream Cloud

Eric Coulter
XSEDE Cyberinfrastructure and Resource Integration Engineer
April 17, 2018
ECSS Symposium
Introduction

• XSEDE Cyberinfrastructure and Resource Integration
• Science Gateways Research Center
• ECSS Consultant
Context

• Natural progression of work with the SGRC
• How to best leverage Jetstream from a Science Gateway perspective?
  – Most gateways expect an HPC-style resource
• Users always want more resources / less queue time – while Jetstream has the space, your lab is the only one in the queue!
Basic VC Design

• Keep It Simple!
• Leverage the OpenHPC project
• Headnode + elastic compute instances
  – How many instances depends on allocation details
  – Typically 5-10; spun up 15 for a tutorial once
  – Login node is optional
• Local user management; federation is an option
• Run jobs either via a Gateway or via ssh session
• The hard part is configuring a researcher’s particular software
What does it look like?

Jetstream

Headnode
slurmctld + Ansible

Router
public ip
private network

compute-0
slurmd

compute-1
slurmd

... 

compute-n
slurmd
Technical Details

• Create Headnode via the openstack cli
• Run installation script on the headnode, creates – working Slurm install, with compute instance create/suspend scripts for elasticity
• Create shared storage if necessary (Cinder volume on the headnode, shared via NFS across instances)
• Node spin-up process
  – Install software, TEST
  – Compute node software controlled via Ansible
  – Slurm actually runs the playbook when node creation is triggered
Technical Details, Pt 2

- Most of the elasticity is handled by slurm (decisions about when to suspend/resume)
- `slurm_suspend` and `slurm_resume` use a mix of the openstack cli and Ansible to handle compute nodes
- This transfers to hardware as well!

```
#CLOUD CONFIGURATION
ResumeProgram=/usr/local/sbin/slurm_resume.sh
SuspendProgram=/usr/local/sbin/slurm_suspend.sh
ResumeRate=0 #number of nodes per minute that can be created; 0 means no limit
ResumeTimeout=900 #max time in seconds between ResumeProgram running and when the node is ready for use
SuspendRate=0 #number of nodes per minute that can be suspended/destroyed
SuspendTime=60 #time in seconds before an idle node is suspended
SuspendTimeout=30 #time between running SuspendProgram and the node being completely down

#COMPUTE NODES
NodeName=compute-[0-1] State=CLOUD CPUs=2
#NodeName=compute-1 Sockets=1 CoresPerSocket=1 State=CLOUD

#PARTITIONS
PartitionName=cloud Nodes=compute-[0-1] Default=YES MaxTime=INFINITE State=UP
```
Existing Production VC’s

• SEAGrid - Sudhakar Pamidighantam at IU
  – Quantum Espresso; 5 nodes, Wrangler mount
• UltraScan - Borries Demeler – UT Health San Antonio
  – Custom UltraScan software; 10 nodes
• NeuroScience Gateway - Amit Majumdar - SDSC
  – Matlab and FreeSurfer; 10 nodes
Benefits

• Researchers benefit from highly configurable compute resources!
  – Your group is the only one in the queue!
  – Researchers can specify exactly what software is available (and are free to build it themselves if needed)

• Workshops/tutorials benefit greatly from flexible infrastructure

• Maximize use of JS allocation for actual compute time
Project Outline

• Researcher gets in touch via some means (so far, typically via contact with the SGRC).

• Discuss problem; ensure that a VC is the right answer
  – Gather requirements – Software, Node sizes (based on compute load needs), number of users, ssh keys

• Add me to the allocation

• Begin the build!
Timeline

• Initial meeting and requirements gathering
  – 1-2 hrs

• Initial VC build
  – 1-2 days (~20 minutes for the minimal version)

• Testing and configuration
  – On the order of a couple of weeks
Sustainability

• Not 100% set-and-forget
  – OS updates
  – JS issues (rare)
  – Slurm gets into a bad state (many reasons, less rare)

• Uses standard HPC components, so diagnosis is relatively “easy”, but not many labs have spare HPC admins lying around.

• Elasticity allows for a bit of self-healing
Questions?

Contact:
help@xsede.org - include XCRI in the subject
xsedeccb@tickets.iu.edu - for more immediate access
Or just @jecoulte in the XSEDE, CC, or Jetstream Slack spaces.

Code is available at:
https://github.com/Ecoulter/Jetstream_Elastic_Slurm