# **Cluster in the Cloud**

#### Easy, Scalable, Heterogeneous



Matt Williams Research Software Engineer University of Bristol

## The problem

• Researchers having cloud credits

### The problem

- Researchers having cloud credits
- Presented with:

≡	Google Cloud Platfor	8• citc		
♠	Home		VM instances	
Ŧ	Pins appear here 🔞	:	Instance groups	
Ŷ	Marketplace		Instance templates Sole-tenant nodes	ue
	Billing		Disks	
API	APIs & Services	>	Snapshots Images	
Ť	Support	>	TPUs	
Θ	IAM & Admin	>	Committed use discounts Metadata	
۲	Getting started		Health checks	
•	Security	>	Zones	
СОМ	PUTE		Network endpoint groups Operations	erv
۰Ô۰	App Engine	>	Security scans	
۲	Compute Engine	>	Settings	
٢	Kubernetes Engine	>		

VPC Dashboard Filter by VPC:
<b>Q</b> Select a VPC
Virtual Private Cloud
Your VPCs
Subnets
Route Tables
Internet Gateways
Egress Only Internet Gateways
DHCP Options Sets
Elastic IPs
Endpoints
Endpoint Services
NAT Gateways
Peering Connections
Security
Network ACLs
Security Groups

aws Services - Re	esource	Groups 🗸 🔪 🗘 matt @ 9634-4975-8405 👻 Ireland 👻 Support 👻				
1. Choose AMI 2. Choose Instance Type	3. Con	figure Instance 4. Add Storage 5. Add Tags 6. Configure Security Group 7. Review				
Step 3: Configure Instance Details						
No default VPC found. Select another VPC, or create a new default VPC.						
Configure the instance to suit your requirements. You can launch multiple instances from the same AMI, request Spot instances to take advantage of the lower pricing, assign an access management role to the instance, and more.						
Number of instances	(j)	Launch into Auto Scaling Group (j)				
Purchasing option	(i)	Request Spot instances				
Network	(i)	Vpc-0865229f3d032b76e   citc-net-cluster-exciting-scdpl C Create new VPC   No default VPC found. Create a new default VPC. C				
Subnet	(j)	Subnet-0bfb83b7ebb34f21d   citc-subnet-cluster-excit Create new subnet   32762 IP Addresses available Create new subnet				
Auto-assign Public IP	(i)	Use subnet setting (Disable)				
Placement group	(j)	Add instance to placement group				
Capacity Reservation	(j					
Open Create new Capacity Reservation						
IAM role	(j)	None Create new IAM role				
Shutdown behavior	i	(Stop 🗘				
Enable termination protection	i	Protect against accidental termination				
Monitoring	()	Enable CloudWatch detailed monitoring Additional charges apply.				
Tenancy	()	Shared - Run a shared hardware instance				
		Cancel Previous Review and Launch Next: Add Storage				

# The problem

- What they already know:
  - Their field of research
  - Python/R/GROMACS/Relion
  - sbatch/qsub
- We can't expect researchers to be professional sysadmins
  - The intersection is well handled by Research Software Engineers

### The solution

- Give them what they are used to, but in a cloud environment
- They don't have to know the difference
- Except:
  - No queuing
  - Only pay for what they use

## **Cluster in the Cloud**

- An automatically provisioned Slurm cluster
- Terraform creates:
  - Networking
  - Shared file system
  - Management/login node
- Ansible configures the management node and compute nodes

#### **Key Features**

- Familiar environment for researchers
- Allows any number of any combination of node types in a cluster
- They are started only when needed, making it cheap to run
- Base cost is just one VM plus storage
- Works on AWS, Google Cloud and Oracle

#### **Technical details: Terraform**

- Terraform is used to create the skeleton
- https://github.com/ACRC/citc-terraform
  - Oracle: ~400 LOC
  - Google: ~250 LOC
  - AWS: ~400 LOC
- Written from scratch for each platform

## **Technical details: Ansible**

- ~1.5K lines of Ansible
- https://github.com/ACRC/slurm-ansible-playbook
- Configures:
  - Mounting shared filesystem
  - LDAP for user management
  - Slurm
    - Including node start/stop scripts
  - Monitoring (Grafana)
  - Base software set
  - And more...
- Covers both the management node and compute nodes

#### Technical details: startnode/stopnode

- Separate Python scripts for each provider
- 160-250 LOC each (+ similar in tests)
- Starts the requested nodes
- Kicks off the bootstrap process
- Sets up networking, DNS etc. as necessary

#### Slurm power management

- Initial configuration creates any number of *potential* nodes of each desired type:
  - e.g. 1000 32-core, 1000 16-core, 1000 GPU etc.
- On job submission Slurm chooses a node
  - It creates a VM
  - Runs the job
  - Destroys it (after a timeout)











#### Node states

• 10 element array job, 5 minute runtime



# Timing

- Full system test ~14 minutes on AWS
  - Create cluster from scratch
  - Submit job
  - Run job
  - Tear down whole cluster
- Job submit → job start: < 2–4 minutes

#### **Performance characteristics**

- Best-suited to heterogeneous high-throughput tasks
  - Pipelines needing different node type for different parts
  - Can be much more specific than the average on-premise cluster
  - Always access to latest hardware
- At present is not optimised for HPC workloads
  - No fast interconnect/parallel filesystem support
  - Future work will rectify this
- Great for teaching clusters
- Suitable for Dask, Spark, Singularity

#### Users

- Bristol:
  - A General Mechanism for Signal Propagation in the Nicotinic Acetylcholine Receptor Family 10.1021/jacs.9b09055
  - Synthetic self-assembling ADDomer platform for highly efficient vaccination by genetically encoded multiepitope display 10.1126/sciadv.aaw2853
- Other universities and private companies too

# Thank you

- Thanks to AWS, Oracle and Google for their support
- Thanks to Chris Edsall for co-developing it
- Thank you for listening
- Follow us on Twitter at @clusterincloud and @BristolRSE