# Cluster in the Cloud

## Easy, Scalable, Heterogeneous

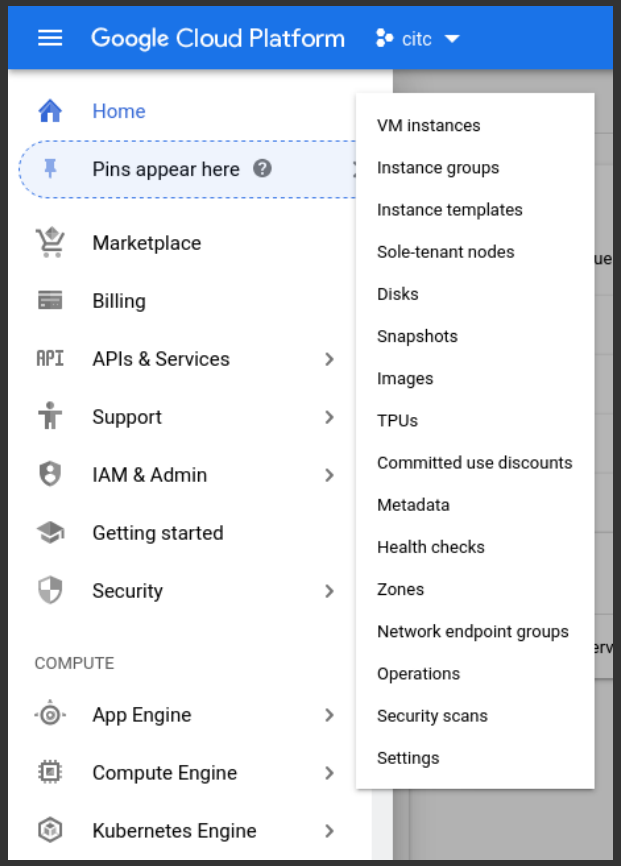

Matt Williams
Research Software Engineer
University of Bristol

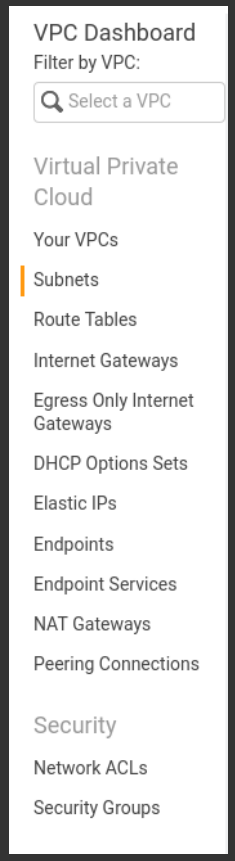
# The problem

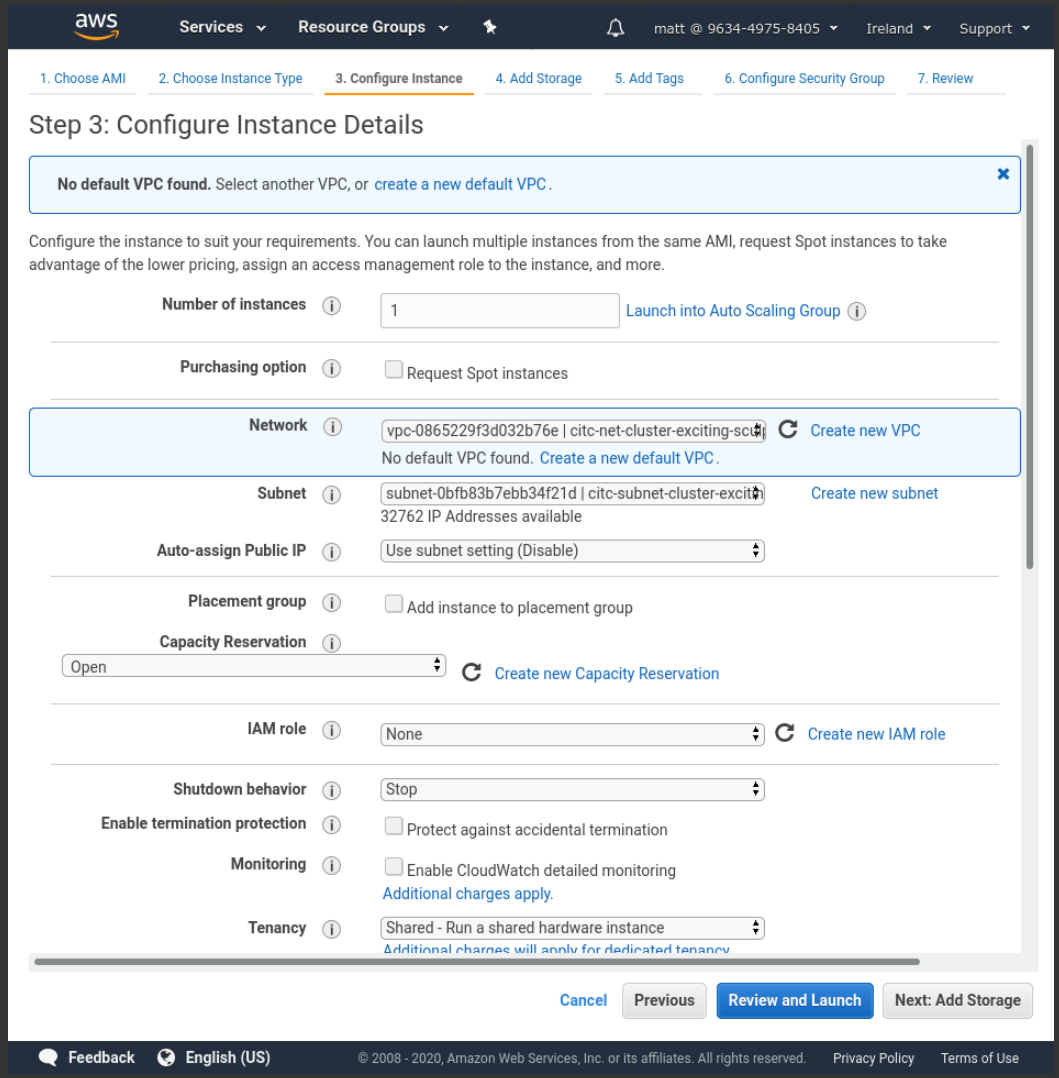
- Researchers having cloud credits

# The problem

- Researchers having cloud credits
- Presented with:



# 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

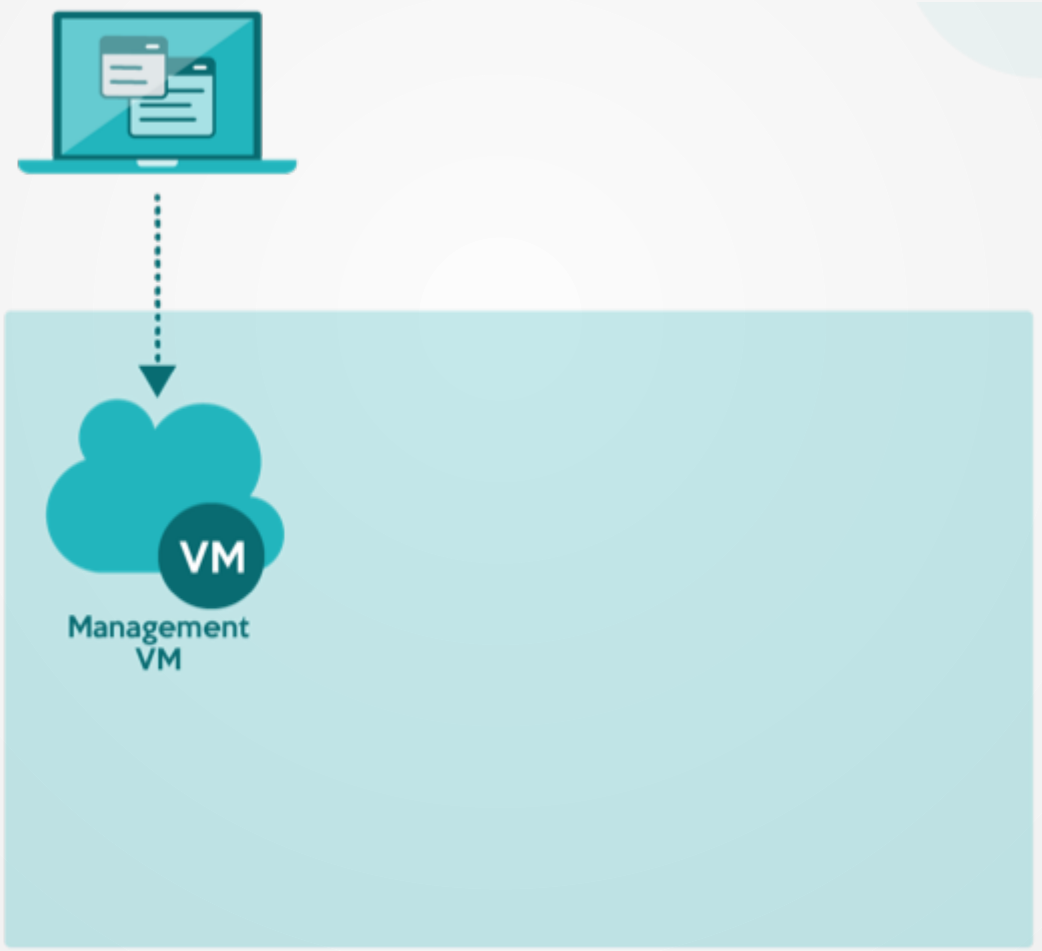
- ~1.5K lines of Ansible
- https://github.com/ACRC/slurm-ansible-playbook
- Configures:
  - Mounting shared filesystem
  - LDAP for user management
  - Slurm
    - Including node start/stop scripts
  - Monitoring (Grafana)
  - Base software set
  - And more...
- Covers both the management node and compute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

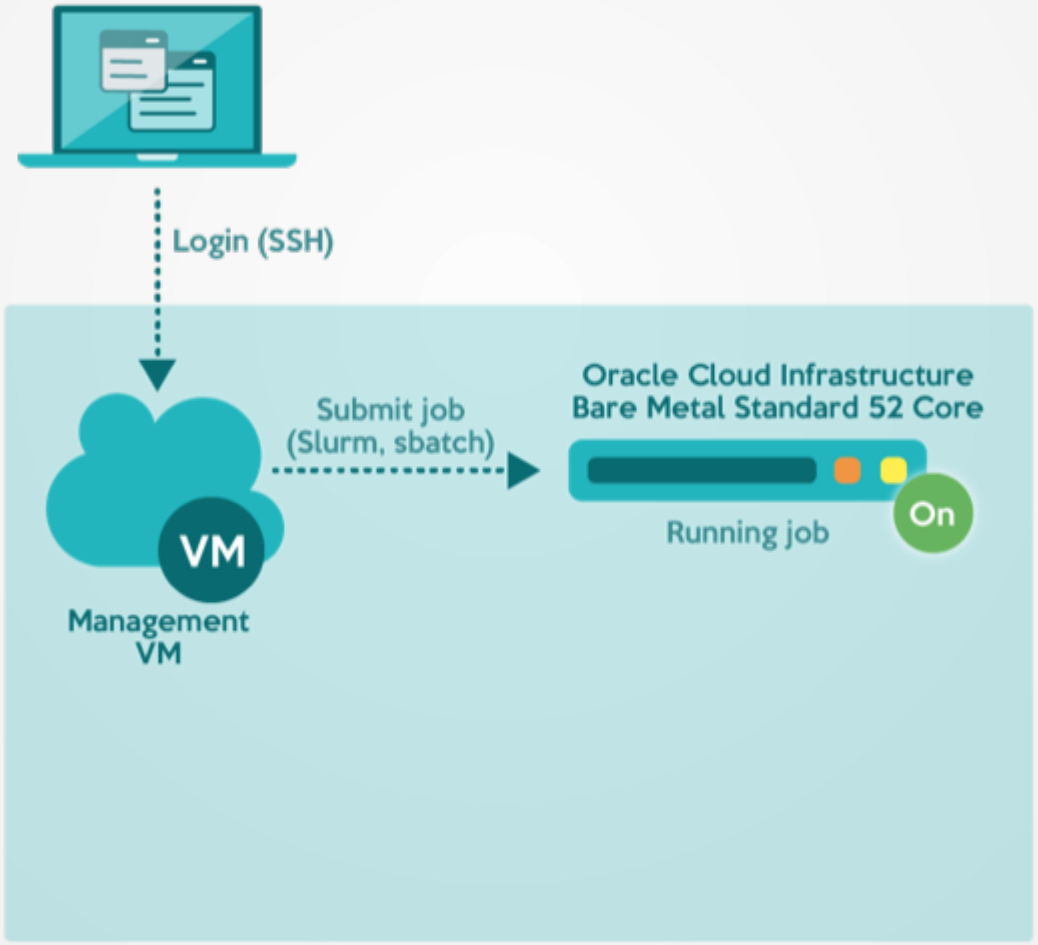
# Technical details: `startnode/stopnode`

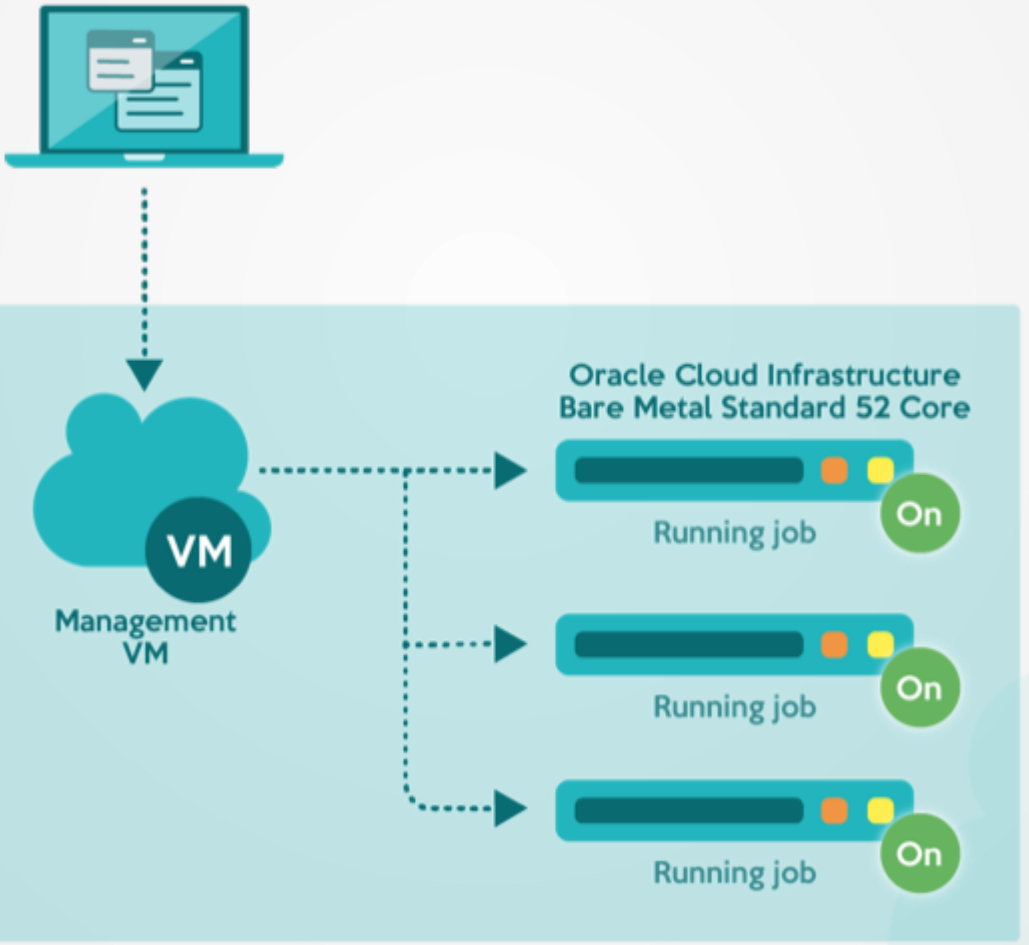
- Separate Python scripts for each provider
- 160-250 LOC each (+ similar in tests)
- Starts the requested nodes
- Kicks off the bootstrap process
- Sets up networking, DNS etc. as necessary

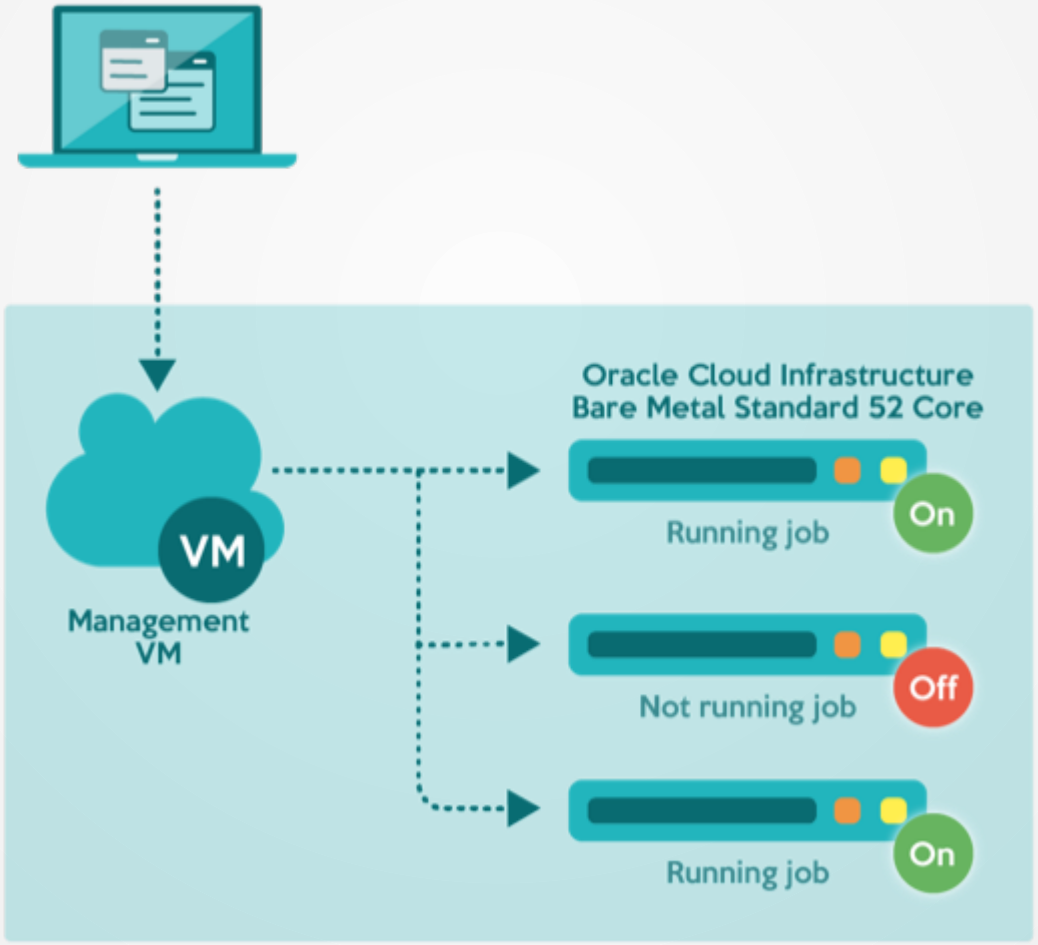
# Slurm power management

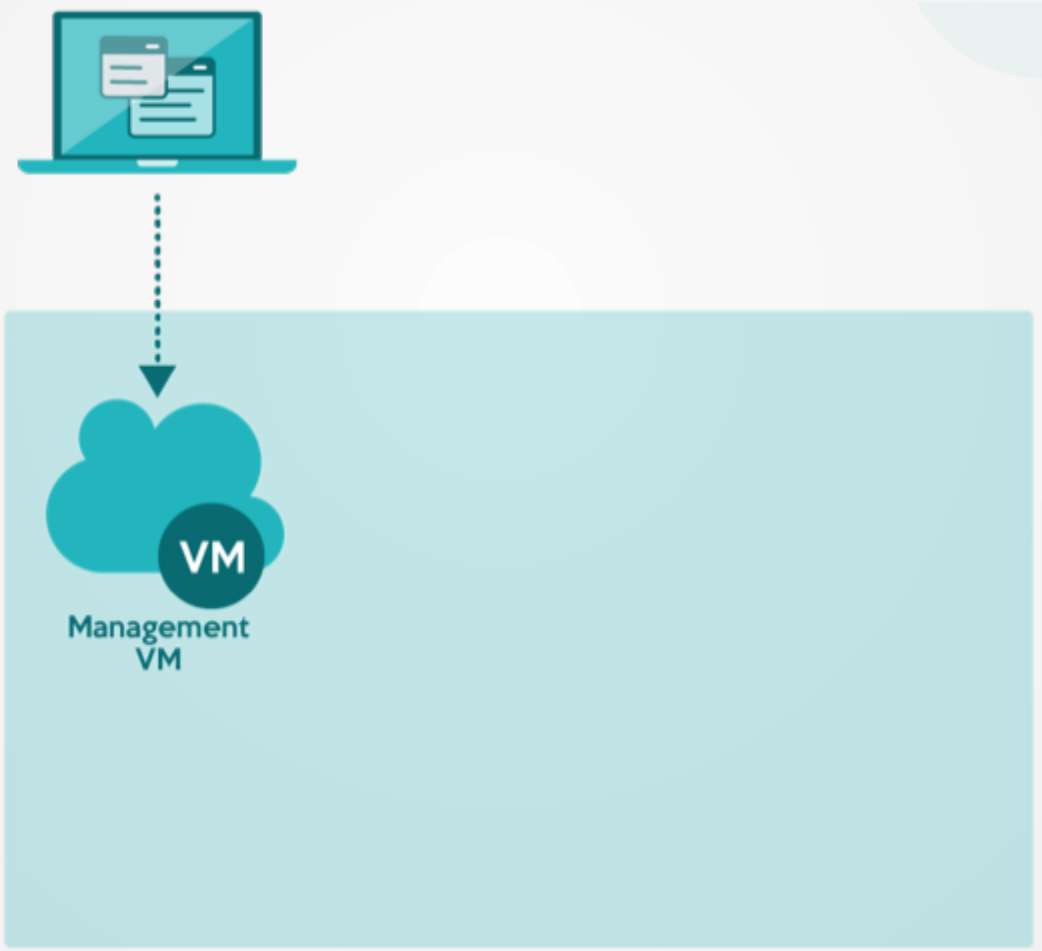
- Initial configuration creates any number of *potential* nodes of each desired type:
  - e.g. 1000 32-core, 1000 16-core, 1000 GPU etc.
- On job submission Slurm chooses a node
  - It creates a VM
  - Runs the job
  - Destroys it (after a timeout)


VM
Management
VM


Login (SSH)
Submit job
(Slurm, sbatch)
Oracle Cloud Infrastructure
Bare Metal Standard 52 Core
VM
Management
VM
Running job
On

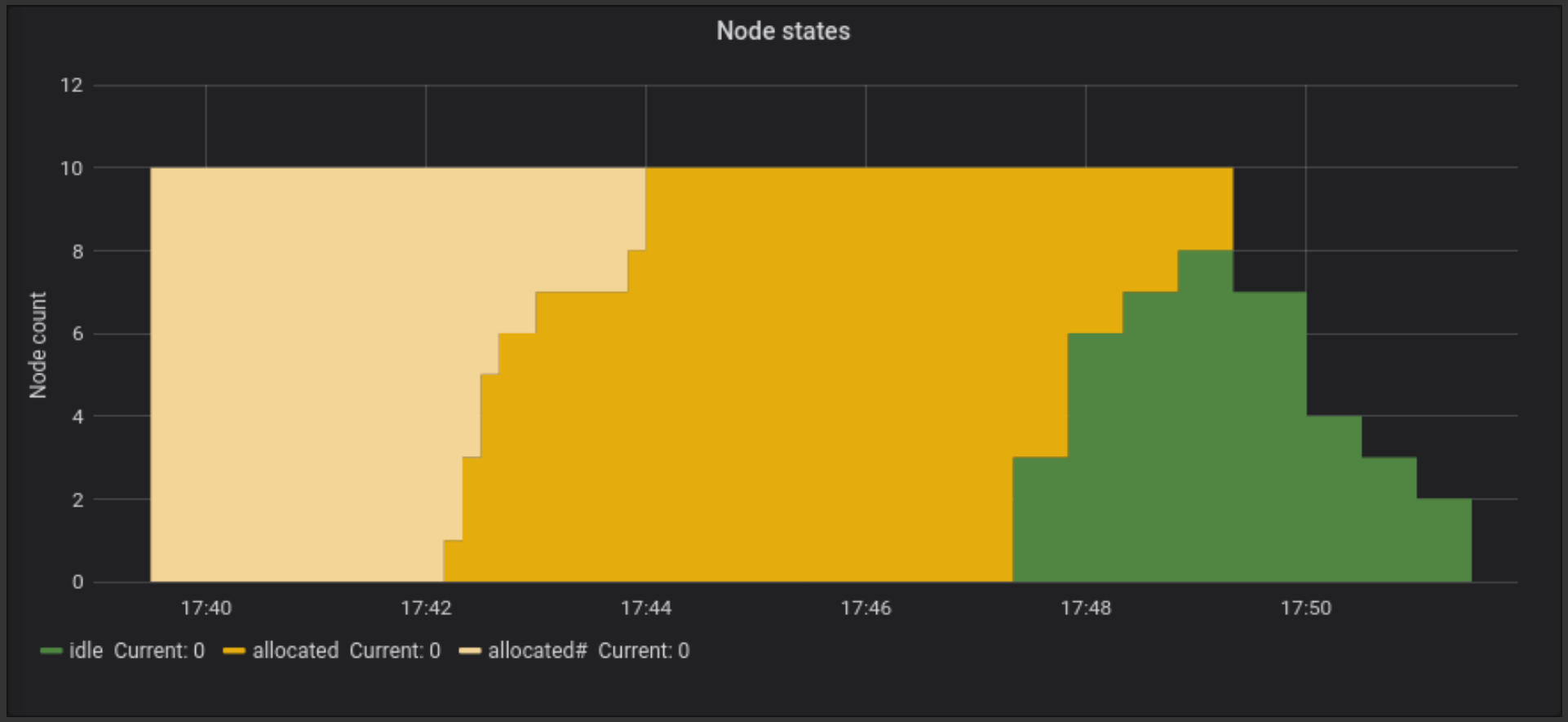

Oracle Cloud Infrastructure
Bare Metal Standard 52 Core
VM
Management
VM
Running job
On
Running job
On
Running job
On


Oracle Cloud Infrastructure
Bare Metal Standard 52 Core
VM
Management
VM
Running job
On
Not running job
Off
Running job
On


VM
Management
VM

# 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