DIRAC & Clouds

Daniela Bauer & Simon Fayer

GridPP
UK Computing for Particle Physics

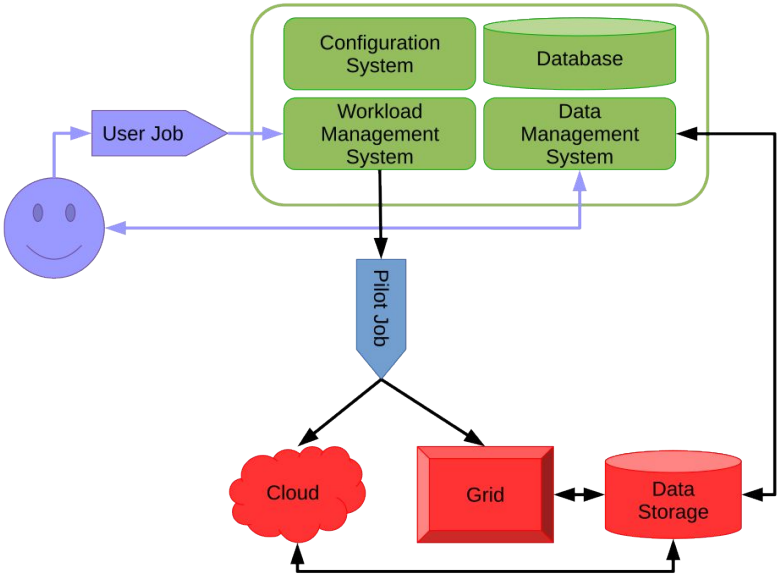
DIRAC

Motivation: Providing large scale distributed compute resources in the UK #worksforgridpp

- GridPP:
  - A collaboration of 18 UK universities and RAL to provide WLCG computing.
  - Also supports non-LHC VOs under a designated “other VO” scheme (~5-10% of resources).
    - GridPP DIRAC was commissioned and is maintained under the GridPP “other VO” remit
  - Resources provided as grid infrastructure (CEs, SEs, WNs, etc).
  - https://www.gridpp.ac.uk/
- **IRIS**:
  - Commissioned by the Science and Technology Facilities Council to provide and coordinate computing resources for UK science facilities (e.g. ISIS, a neutron and muon source), astronomy, nuclear physics, neutrino physics (e.g. DUNE) and others.
  - **GridPP is an IRIS partner.**
  - **Resources are mostly provided as clouds.**
  - **IRIS funds “Digital Assets”: Software projects of limited duration to produce an agreed product.**
  - https://www.iris.ac.uk/

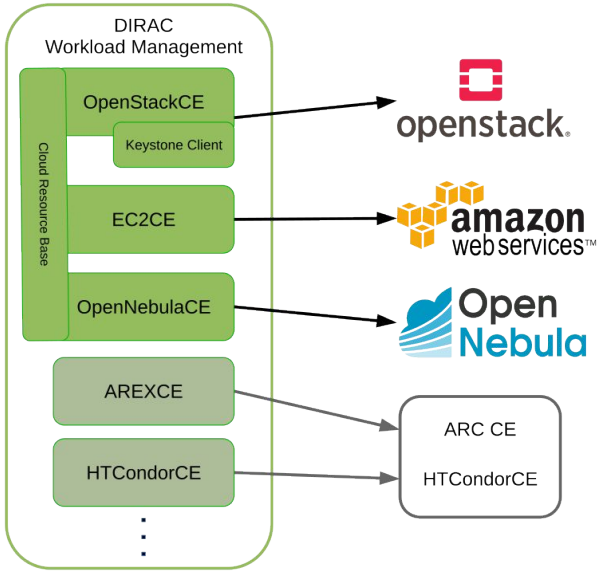
DIRAC as a Cloud Interface

- DIRAC was always foreseen as a cloud interface, shielding the users from the details of different cloud implementations.
- Cloud support was originally provided by a self contained extension, maintained separately to DIRAC.
- Cloud support was reintegrated into DIRAC in 2022, using Apache libcloud as its backend interface library (IRIS digital asset product).



DIRAC Cloud support - original implementation (VMDIRAC)

- One plugin per cloud type.
- Custom contextualisation system
  - updated to cloud-init in 2019
- Initially it was assumed workloads would run on commercial clouds, using fixed allocations, hence there was a notion of pricing and reservations.
- Large code base, not all of which was maintained by 2019.



But what about VCYCLE ?

VCYCLE (https://www.gridpp.ac.uk/vcycle/) was a GridPP/LHCb standalone project that allowed clouds to **pull** (no pilots!) jobs from a DIRAC instance using the vacuum model to generate VMs.

Clouds using VCYCLE required little configuration in DIRAC. Impossible to debug though.

Successfully deployed on a number of sites.

Sounds good-ish, so what happened ?

Single developer project, developer moved on, no takers, project fell into disrepair. (Last git commit in Dec 2019. Last successful GridPP jobs in Dec 2021.)

Lesson: Best laid plans don't work unless there's a critical mass of people behind it. (Don't write code in a vacuum ?)

So, can we lessen the maintenance burden ?

- Reuse, reuse, reuse.
- Take the DIRAC concept of a CE and Apache libcloud, combine and only add the missing link.
- Profit ?

DIRAC: ComputingElements

- DIRAC uses a resource specific plugin known as a ComputingElement (CE) to manage its pilots.
- These plugins implement either a local or remote interface depending on what is appropriate for their target resource:
  - Remote: HTCondorCE, ARC and SSH (primarily used for direct access to HPC resources)
  - Local implementations run inside the pilot and perform tasks such as containerization. (SingularityCE) and multi-core slot partitioning (PoolCE).
- To make this concept work for clouds, each virtual machine is treated as a single pilot job, which is processed by DIRAC **in the same manner** as pilot jobs sent to grid resources.

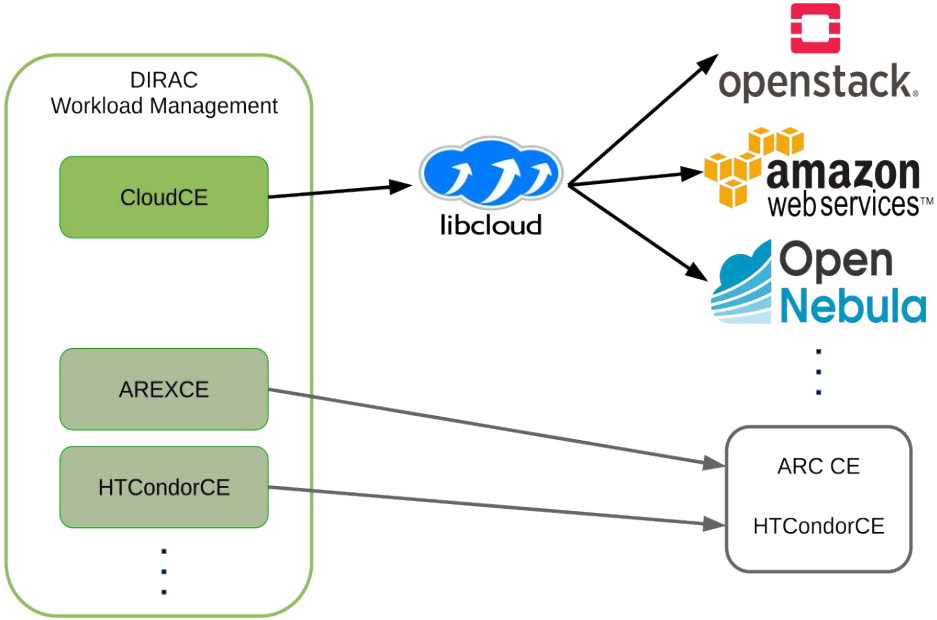
Apache libcloud

https://libcloud.apache.org/

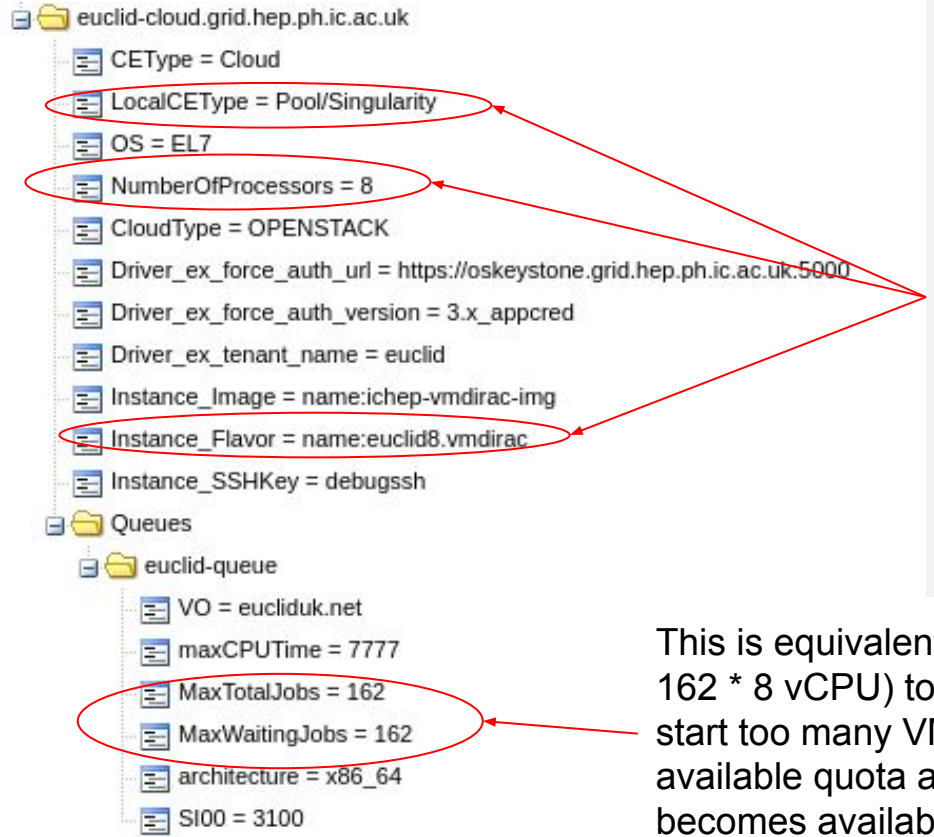
- Apache libcloud is an open source collection of python based cloud interfaces, maintained by the Apache foundation.
  - Cloud compute and storage is supported, but we currently only focus on compute.
- New releases approximately twice a year.
  - We provided one patch for Openstack Application credentials, but otherwise use code as is.

CloudCE: Not so special anymore.

- Inherits from DIRAC ComputingElement
- Instead of communication with a grid compute element, the code calls the respective libcloud interface with the correct parameters/credentials
- The pilot payload script and data are added as instance metadata in cloud-init format; this allows any image containing cloud-init to decode and start the DIRAC pilot bootstrap scripts.
- **We pride ourselves in LOC removed :-)**



Example of a working configuration



This setup (NumberOfProcessors = vCPUs in Flavour):

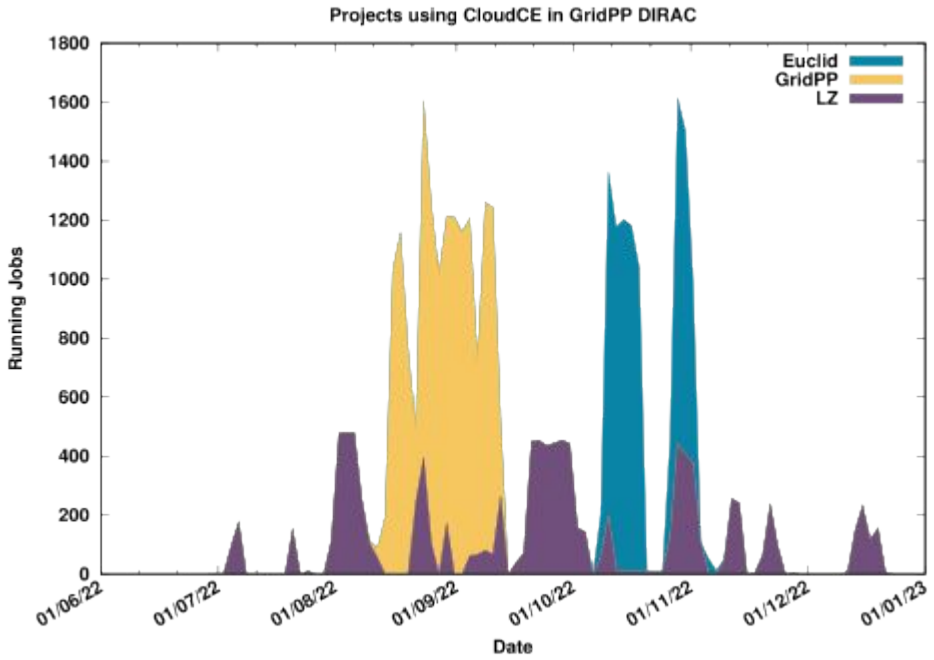
- minimizes IPv4 footprint (PoolCE):
  - initially envisaged using private project networks
  - as cloud usage increased our NAT became overloaded resulting in dropped connections
  - started using provider network directly (bridged public IP address for each instance)
- minimizes pilot code downloads (PoolCE)
- isolates the pilot (Singularity)

This is equivalent to the project quota on the cloud (1296 = 162 * 8 vCPU) to avoid overloading the cloud by trying to start too many VMs at the same time. If more jobs than the available quota are waiting, DIRAC will hold them until a VM becomes available.

It’s working !

Documentation:

https://dirac.readthedocs.io/en/latest/CodeDocumentation/Resources/Computing/CloudComputingElement.html



GridPP
UK Computing for Particle Physics

DIRAC

Other cloud considerations

- Proxies:
  - DIRAC uses the proxy renewal mechanism provided by the CEs for the pilot proxies. In clouds the necessary inbound external connectivity cannot be guaranteed.
  - Therefore DIRAC needs to create a sufficiently long-lived pilot proxy when starting the VM.
  - User proxies are renewed by the DIRAC pilot agent as usual and do not need special consideration for a cloud service.
  - This problem is not going to go away with tokens (please tell me I'm wrong ?)
- Pilot logs: The lack of inbound external connectivity also has implication for the retrieval of pilot logs, which is sometimes necessary for debugging. This is being addressed by changing the pilot logging to a push model.
- So far we had no use case for cloud storage:
  - libcloud officially supports:
    - Cloud Object Storage and CDN - services such as Amazon S3 and Rackspace CloudFiles
  - DIRAC has an S3 storage class: https://dirac.readthedocs.io/en/latest/AdministratorGuide/Systems/DataManagement/s3.html
  - But: no use case, no funding, no code

Where do we go from here ?

- It’s all OpenStack ?! Yes. We do not have access to anything else. If you are using a different cloud type, please let us know *now* and hopefully we can test it during the workshop.
  - One request for OpenNebula so far. Simon & Igor are working on it.
  - If you have an OpenStack cloud and want to test it, this is a three step process:
    - Make an account for the dirac pilot following whatever procedure you have to create accounts.
    - Let your favourite DIRAC admin know the login details.
    - They will then make the application credentials, configure your site and send a test job.
    - If the documentation isn’t clear, please let us know.
  - If you have any other cloud type, please let us know, we are here until Friday.
- We intend to support this component of DIRAC for the foreseeable future.