CMS job submission to HPCs

Dirk Hufnagel- FNAL, Christoph Wissing - Desy, Daniele Spiga -INFN
A. Pérez-Calero Yzquierdo - CIEMAT,PIC

Joint Operations for HPC and Opportunistic Resources
CERN 5th Jan. 2025

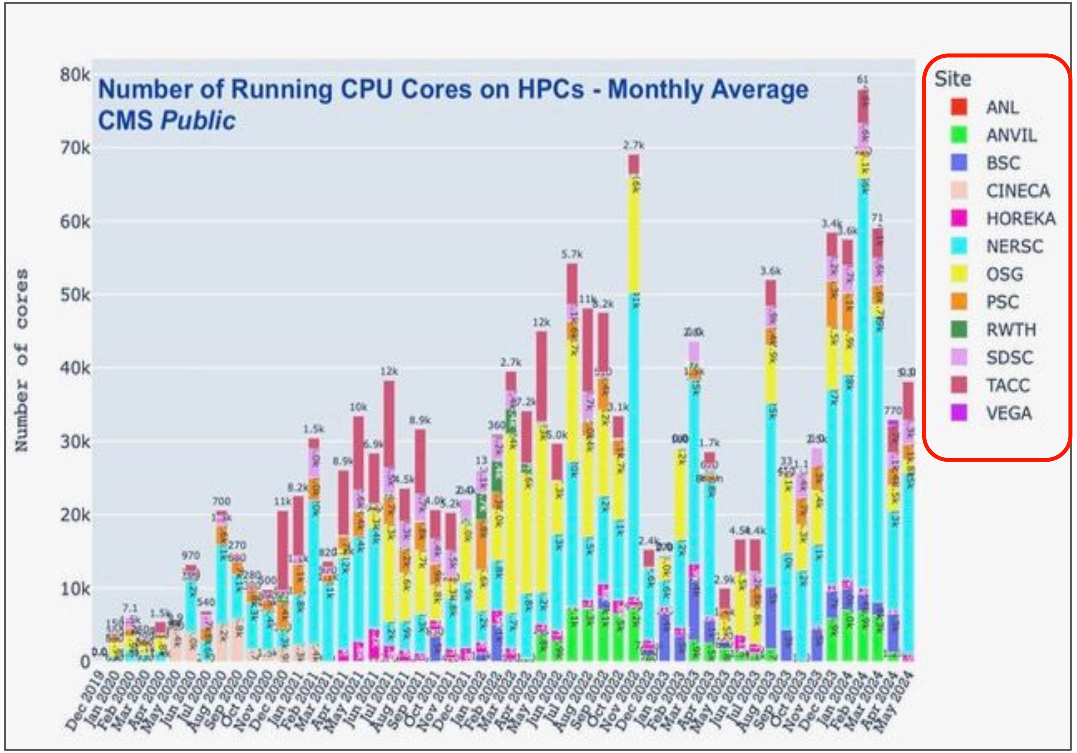
HPC @ CMS

Continuous effort is spent to integrate HPC resources to increase HPC contributions

- CMS wants to further increase the HPC exploitation particularly in the EU zone where CMS uses less resources compared to the US

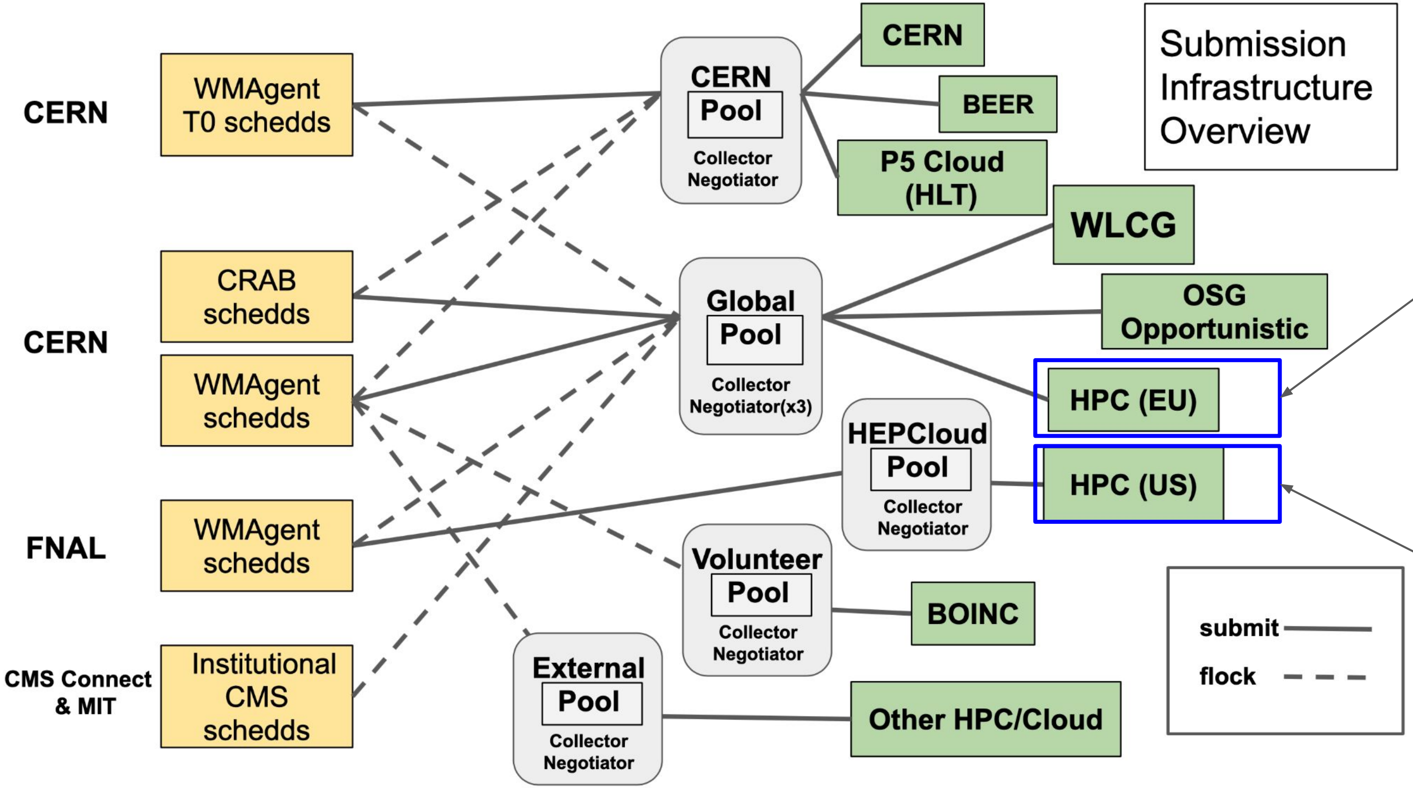
CMS Computing / local teams have to provide interfaces between experiment and HPC

- No one-fits-all solution for HPC integration.



Parts of the pledges are contributed by fully transparently integrated HPCs - i.e. CSCS, CINECA,

General overview of HPC resources integration in CMS



In general, EU HPCs are integrated via the main CMS (Global) HTCondor pool, making them extensions of existing sites at DE, IT, ES.

US HPCs are managed from FNAL, via a dedicated HTCondor pool (HEPCloud) which is federated with the main CMS Global pool

Glideins and GlobalPool: the common trait

All our integrations have peculiarity in the resource provisioning approach (aka how they start glideins). The common trait is that all them bring resources (slots) into the CMS GlobalPool

- Via regular GlideinWMS push model (i.e via CE)
- Via schedd flocking
- Via manual started glideins at site
- …

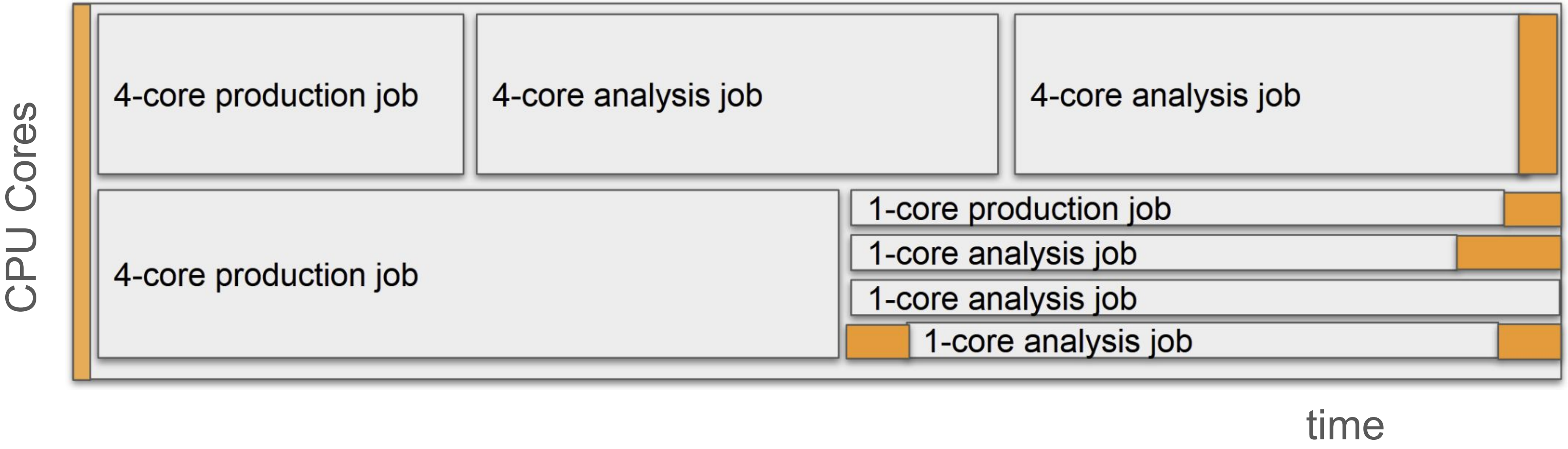
Regarding the Job slots there is no enforcement on a particular size from the CMS side. We can configure depending on site specific needs/constraints

- Wholenode and then glidein-htcondor does the partitioning based on cores/mem/disc
- Multicore (8 cores) slots

Multicore pilot model in CMS SI

HTCondor partitionable slots allow CMS to execute multiple payload jobs concurrently and consecutively for the duration of the pilot lifetime.

Scheduling of individual payload jobs into the resource slots is managed by CMS (late binding).



Workflow Setup

CMS targets to make the HPC i**ntegration as transparent as possible** and tries to run almost **all production workflows at HPC**, but often we cherry-pick the "easiest" (least demanding in terms of I/O) types

- (Too much) cherry-picking has a cost on central ops at CMS

As a matter of fact

- We focus on MC workflows (vast majority of our work, StepChains well suited for HPC)
- More relaxed where the HPC integration is implemented as site extension
    - Site extension based integration also means “direct link” with storage at reference Tier-1
    - Custom matching rules (defined at site) to filter out non suitable workflows (“merge” etc)
- For HPC as a separate site (BSC, HEPCloud HPC) CMS controls assignments of workflows
    - BSC limited in what workflows can run (due to the integration constraints)
        - Longterm goal to be able to run ‘everything’ and switch to T1 site extension
    - HEPCloud HPC can run almost all MC workflows (and is normally auto-assigned for them)

Data Delivery

Input data handling

- Mainly rely on AAA, the CMS xrootd federation
  - StepChain (Gen to Reco) MC workflows need to read pileup from remote storage
  - In some case we use xcache (through proxy mode) to fan-out
- In cases of a preferred ‘close’ storage site, that is the first try and AAA is fallback
  - Obvious for site extension, but even HEPCloud HPC use this mode (FNAL)

Output data handling

- Produced output goes to grid storage site
  - Extended sites relies on storage of the reference site (plus fallback)
  - HEPCloud HPCs in the US all stageout to FNAL directly

Additional considerations

We see the possibility to deploy edge services as a key element to improve.
It seems to be more popular now at HPC sites. These are instrumental and ease the integration with CMS infrastructure

- (self) deployed “custom” services at the boundary of HPC simplify the integration
  - Some early experiences with edge services to provide cvmfs/frontier at LCF in US
- Allow to standardize
- Open the doors to additional workflow (ML/AI and GPU access in general)
  - Prototyping interLink, a edge service enabling payload offloading i.e. from Analysis Facility at INFN

HPC with limited or restricted network won’t support remote data read use case well / at all

- Will need storage allocation at HPC and active data management
- Could also use this for local stageout (remote stageout through edge services also possible)

BACKUP

Overlay batch && site extension

German HPC Resource Integration using `COBalD/TARDIS`

Layers of Abstraction:

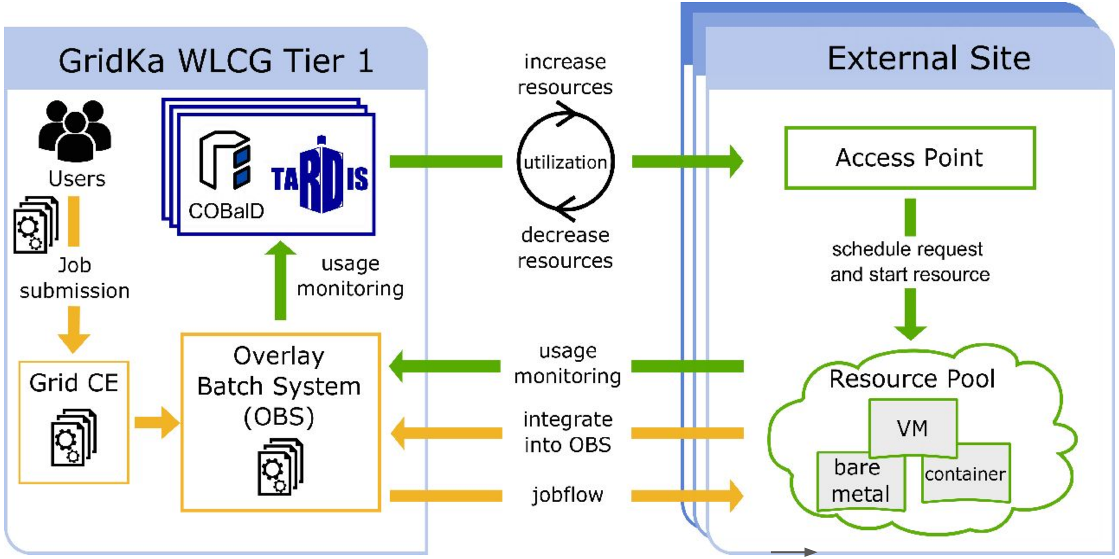
- Grid Compute Elements act as well established single point of entry
- Overlay Batch System provides a single pool of resources hiding complexity

⇒ Transparent & hassle-free access to HPC resources

Resource Integration:

- Utilizing **`COBalD/TARDIS`** resource manager
- Simple feedback based approach: “More used, less unused resources”

⇒ Efficient and dynamic access to HPC resources



Integration of CINECA into CMS Computing

CMS and CINECA were able to agree (2019) on a minimal set of changes to allow for CMS job processing.

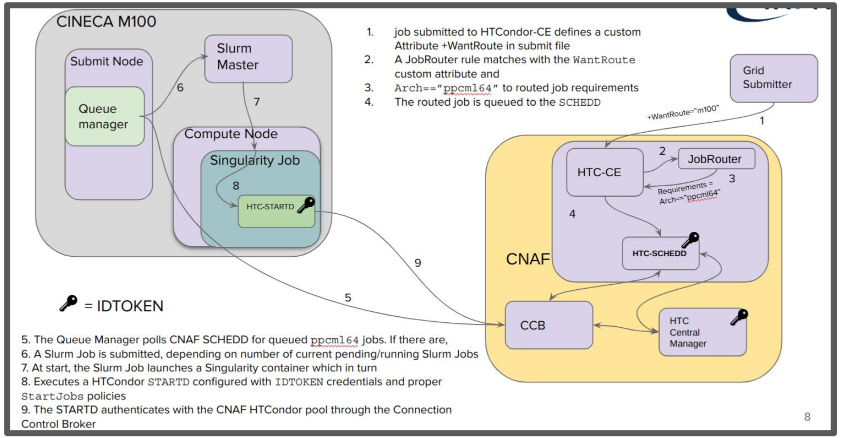
- CVMFS was installed on the systems; Network routing to CERN and CNAF IP ranges activated; Singularity audited and deployed;

CINECA nodes were configured as **an elastic extension of CNAF Tier-1** (SubSite concept) receiving all the jobs targeted for the standard WLCG site.

- Input data access to the AAA Data Federation via xrootd proxy (@CNAF)
- Worker Nodes provisioning: via site launched glidein Relying also on custom matching rules (defined at site)
- A cherry-picking method has been adopted allowing site-level specification of additional requests with respect to CNAF nodes in order to select most suitable workflows.

Same setup used also to integrate VEGA, a transnational site extension

Prototyping also a slightly evolved model transparent T1 batch extension



1. job submitted to HTCondor-CE defines a custom Attribute +WantRoute in submit file
2. A JobRouter rule matches with the `WantRoute` custom attribute and
3. `Arch=="ppcml64"` to routed job requirements
4. The routed job is queued to the `SCHEDD`

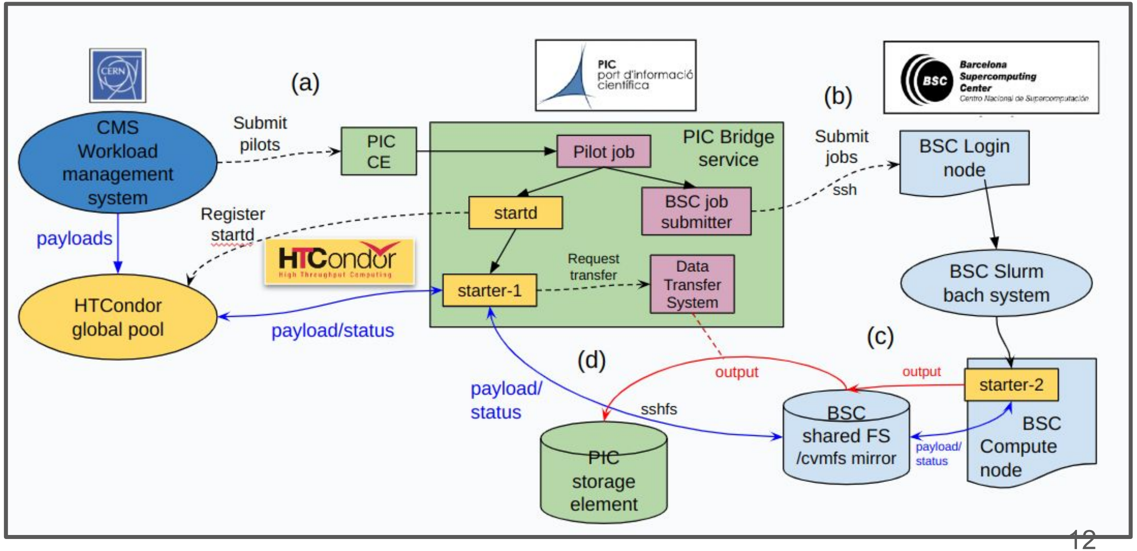
5. The Queue Manager polls CNAF SCHEDD for queued `ppcml64` jobs. If there are,
6. A Slurm Job is submitted, depending on number of current pending/running Slurm Jobs
7. At start, the Slurm Job launches a Singularity container which in turn
8. Executes a HTCondor `STARTD` configured with `IDTOKEN` credentials and proper `StartJobs` policies
9. The STARTD authenticates with the CNAF HTCondor pool through the Connection Control Broker

Barcelona Supercomputing Center (BSC)

- **Very restrictive network connectivity conditions**
  - No incoming or *outgoing* connectivity from compute nodes
  - No services can be deployed on edge/privileged nodes
  - Rely on ssh to login node and sshfs to internal shared FS

Technical solutions

- HTCondor split-starter: use shared filesystem as communication layer for job management
- Software: replicate CVMFS repositories at BSC storage (CMSSW, singularity images, etc)
- Detector conditions data files: pre-placing them at BSC not feasible, reverse ssh-tunnels access to PIC squids
- Developed a service for in/output data transfer
- PU datasets need to be manually copied into BSC gpfs in order to run full simulation sequence



FNAL HEPCloud for US HPC

Based on GlideInWMS

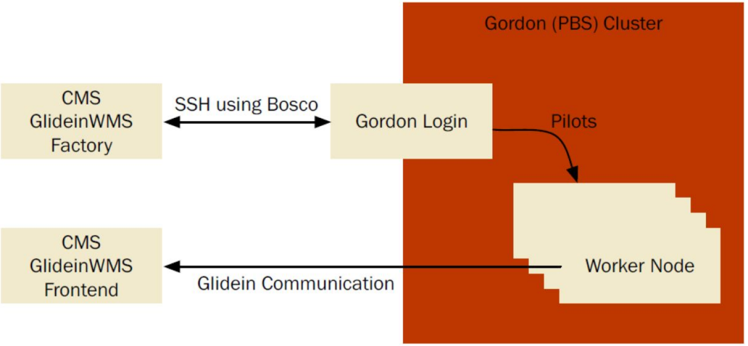
(underlying that HTCondor)

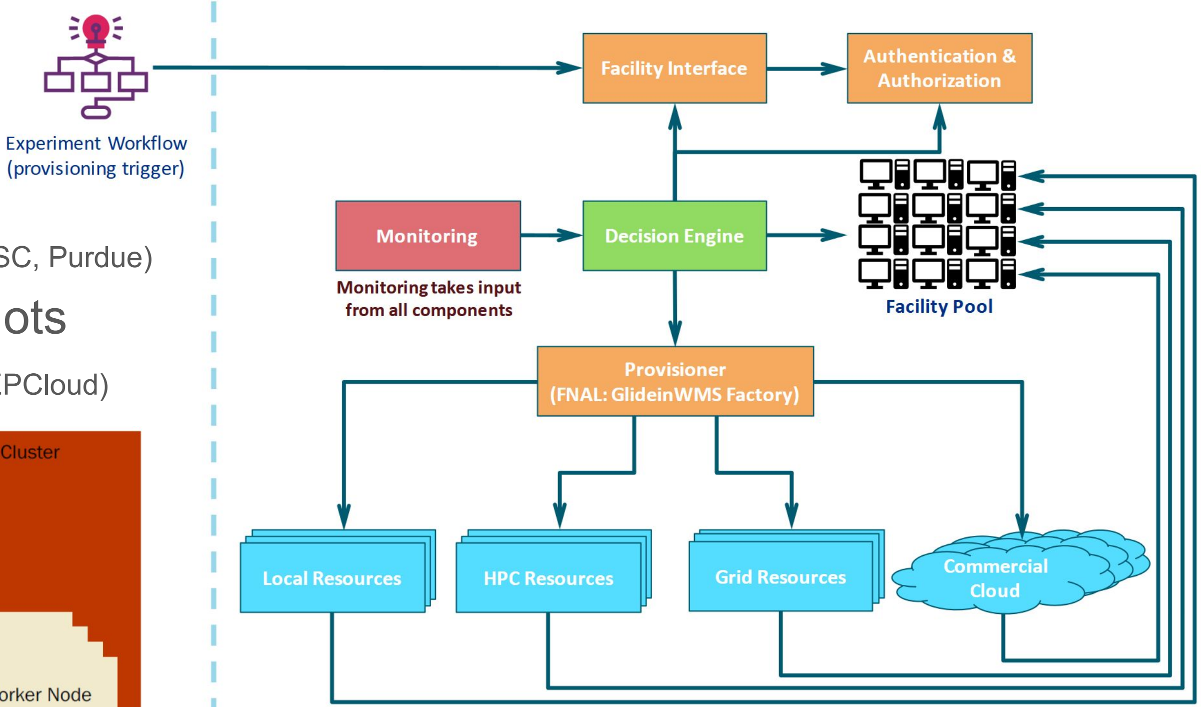
Interface to multiple HPC

NERSC, TACC Frontera, ACCESS (PSC, SDSC, Purdue)

Remote ssh submission of pilots

First used with SDSC Gordon in 2014 (pre-HEPCloud)




Experiment Workflow (provisioning trigger)
Facility Interface
Authentication & Authorization
Monitoring
Monitoring takes input from all components
Decision Engine
Facility Pool
Provisioner (FNAL: GlideinWMS Factory)
Local Resources
HPC Resources
Grid Resources
Commercial Cloud