Centro di Competenza sul Calcolo Scientifico

Provisioning flessibile di risorse di calcolo con OCCAM

Marco Aldinucci, Paolo Pasteris, Sergio Rabellino

Department of Computer Science and C3S, University of Torino

Stefano Bagnasco, Stefano Lusso, Sara Vallero, Matteo Concas

Istituto Nazionale di Fisica Nucleare, sezione di Torino and C3S, University of Torino

Users

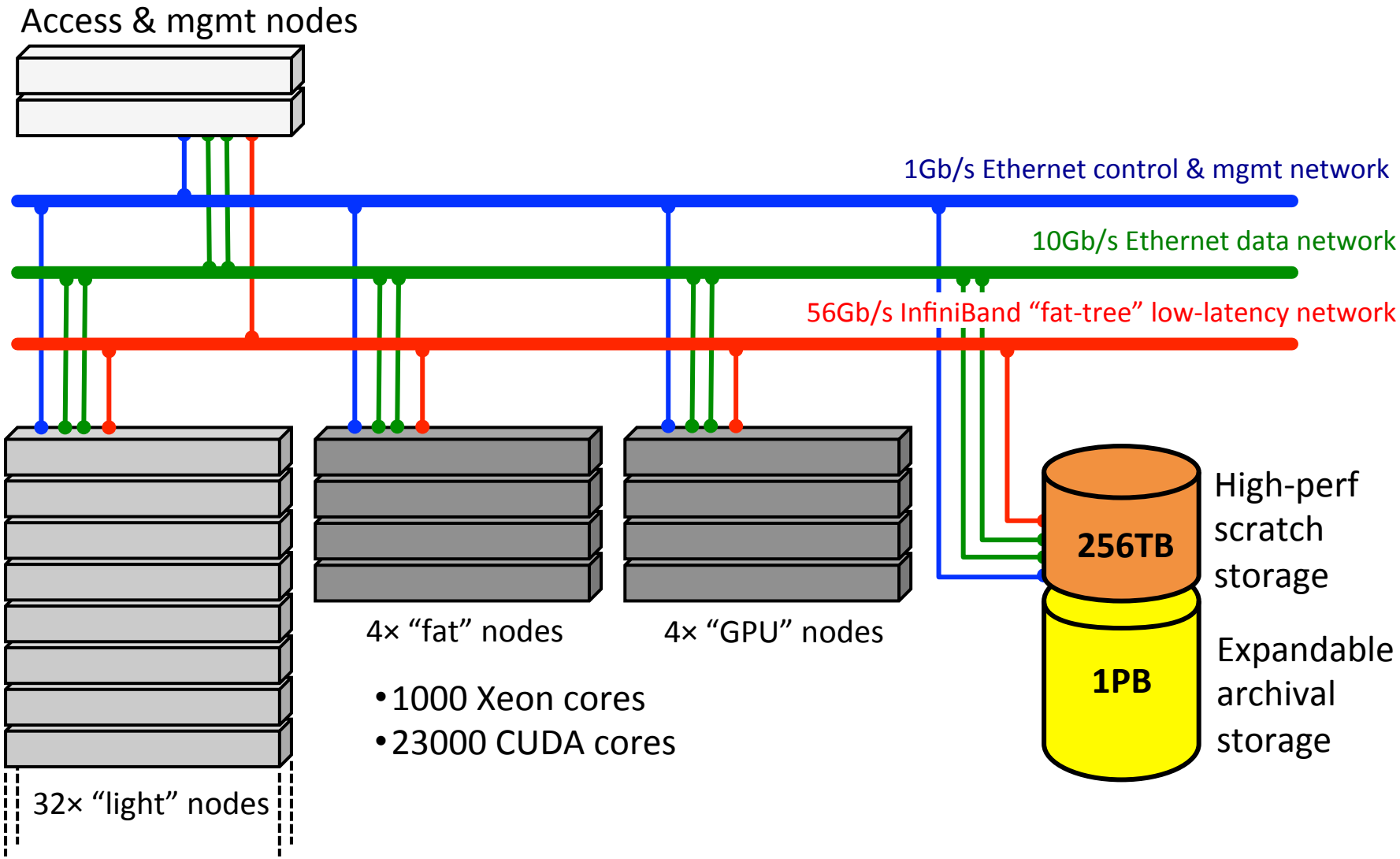
- A very large array of scientific use-cases from 18 university departments:
  - Computational chemistry
  - Genomics, transcriptomics & other -omics
  - Complex systems in several disciplines
  - HEP (and more) code testing & porting
  - Pharmacology & drug discovery
  - Big Data in economics & the social sciences
  - ...you name it, we have it.

OCCAM

Open Computing Cluster for Advanced data Manipulation

Compagnia di San Paolo

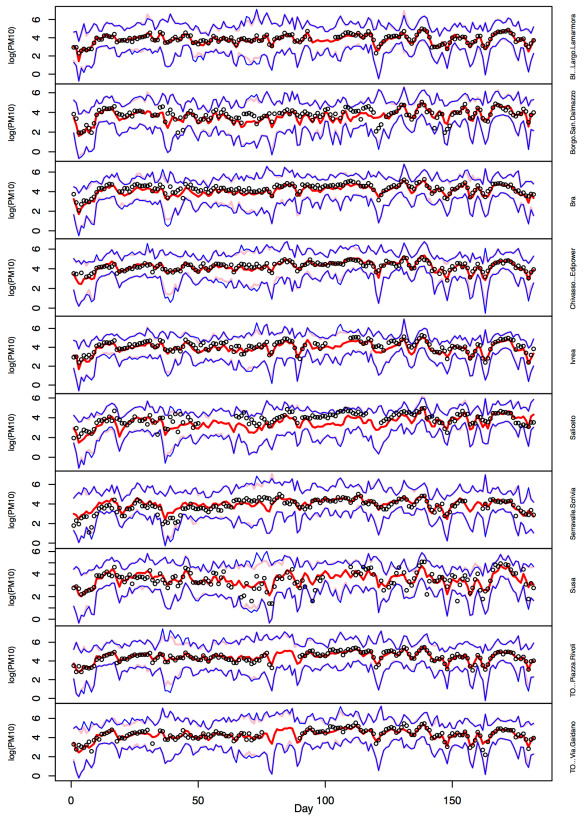
Architecture



- 1000 Xeon cores
- 23000 CUDA cores

Use case 1

- Environmental data monitoring and forecasting
  - A team at the Statistics department analyses atmospheric NO_2 data using air quality data and numerical transport models.
  - R-based code uses a bootstrap technique that requires repeated access to a relatively large amount of data. Computational power and memory requirements are moderate.
  - Several such use cases do exist, typically R- or python-based code that could run on a single large workstation.



Use case 2

- Ab-initio Solid State Chemistry

  - CRYSTAL is a widely-used software for computational chemistry maintained by a team from the Chemistry Department of the University of Torino
  - The code is developed since the 1970s, and can be applied to the study of any type of crystalline material, with a special focus on the simulation of vibrational spectra.
  - The MPI code does not have huge memory requirements and scales well to thousands of parallel cores, so they need a large number of HPC cores, with little or no need for data access.

Use case 3

- Classification Analysis of Single Cell sequencing data
  - CASC is a Computational Biology software for Classification Analysis of Single Cell sequencing data developed by a from the Biotechnology and Computer Science Departments
  - The code is R-based and is distributed as a set of Docker containers that run in sequence, each using the output of the previous one.
  - Because of the large memory requirements and data access patterns, the software does not scale to more than a few cores, and needs relatively high bandwidth access to data storage.

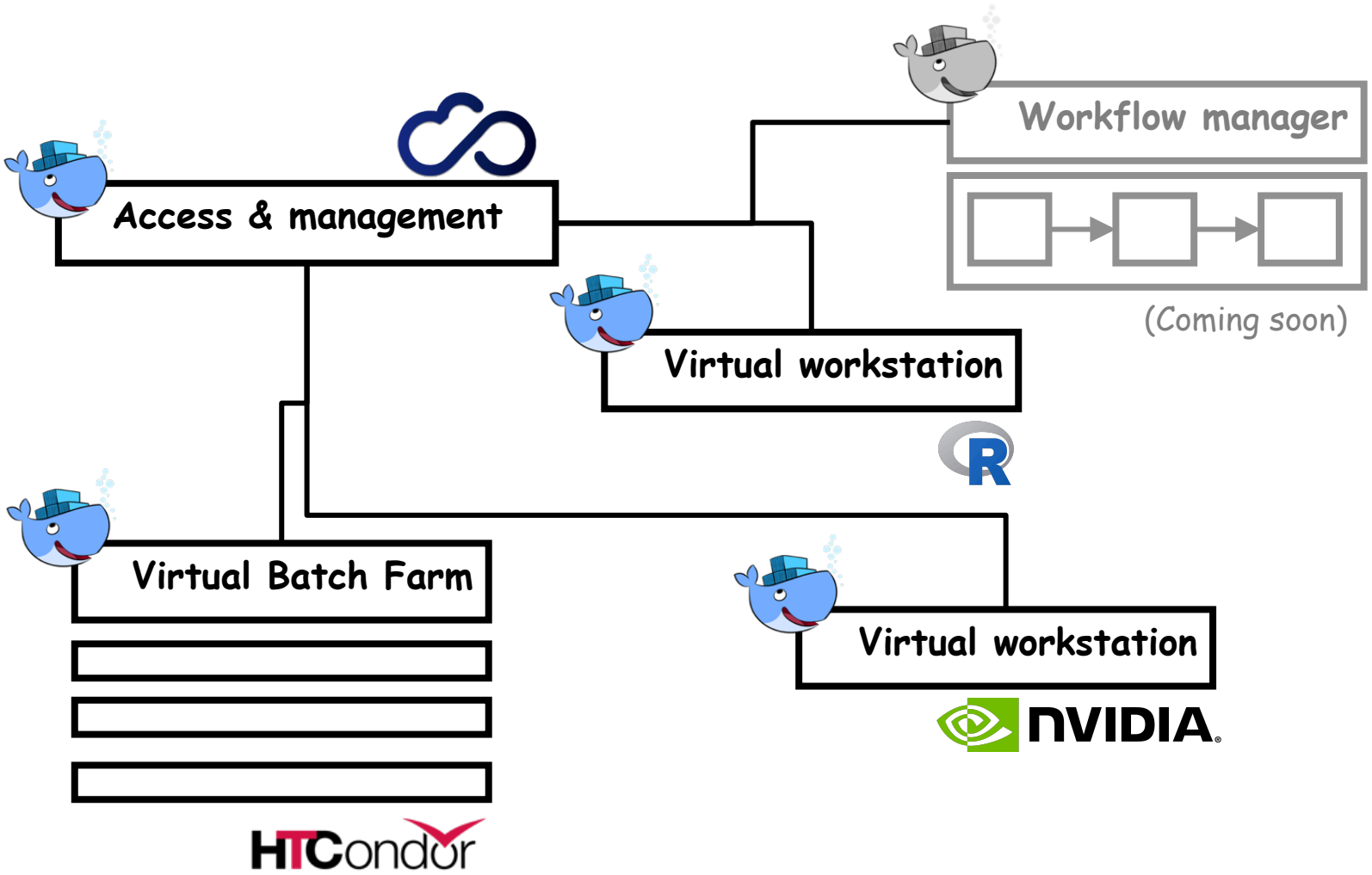
Borrow some Cloud Computing ideas and adapt them to an HPC environment

- Instead of setting up a batch system and run batch jobs, we run "Computing Applications"
- A Computing Application is defined by its **runtime environment**, its **execution model** and **resources requirements**
- Each Computing Application is granted use of an isolated virtual cluster, so it sees only the resources it is allowed to use

Virtual workstation: batch or interactive code execution (e.g. R or ROOT) in a single multicore node, possibly with GPU acceleration

HPC: batch-like, multi-node workloads using MPI and inter-node communication

Genomic pipelines: multi-step data analysis requiring high-memory large single-image nodes


Workflow manager
(Coming soon)
Access & management
Virtual workstation
Virtual Batch Farm
Virtual workstation
NVIDIA
HTCondor

Toolkit

Docker: Industry-standard containerization platform

Used to partition the system into **isolated virtual clusters** to run Computing Applications.

Also, self-packaging decouples infrastructure from application software management

Apache Mesos: resource abstraction and management

Mesosphere Marathon: long-running services scheduling and monitoring

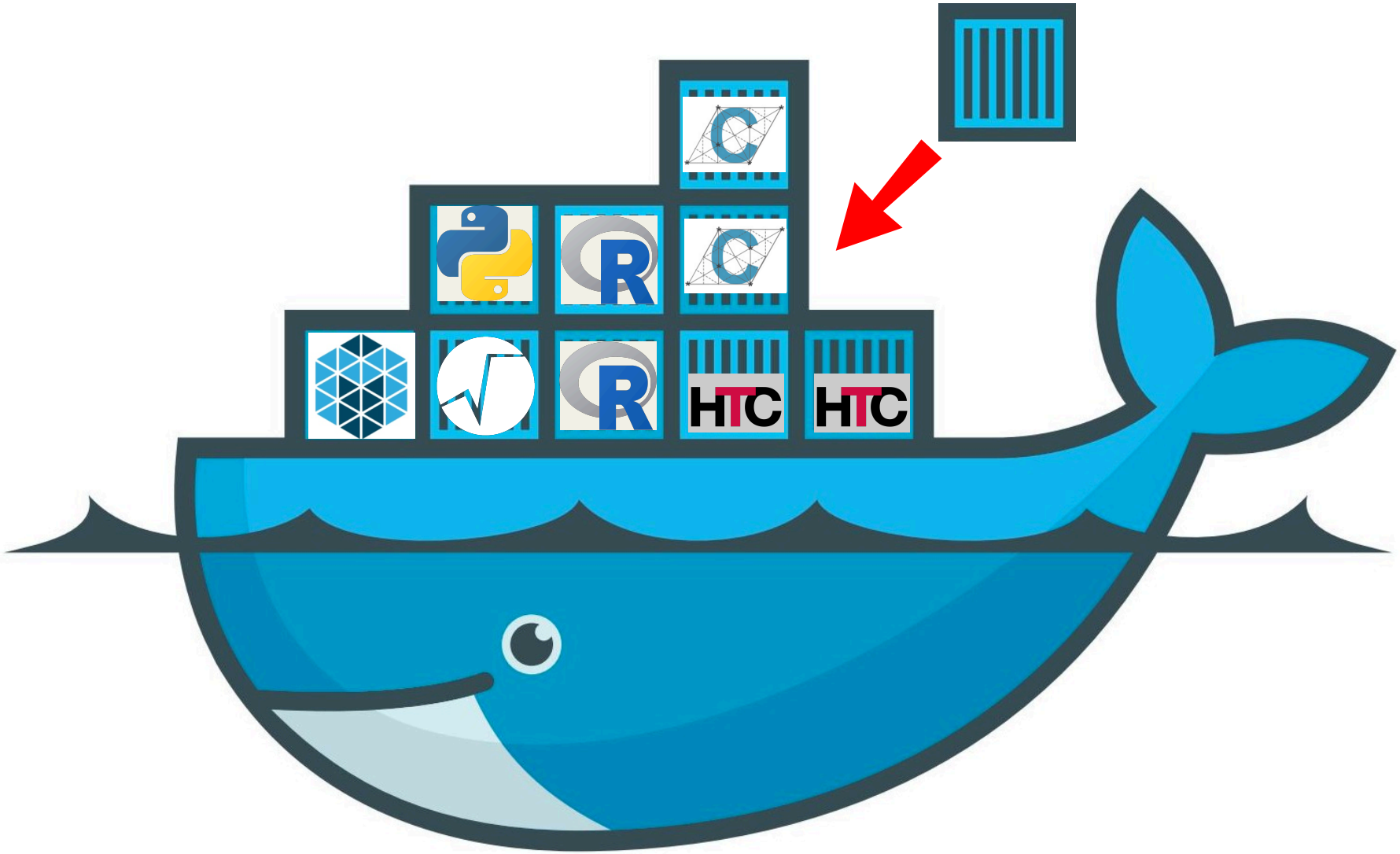
Used to **schedule, deploy and manage** Computing Applications

Calico: enable secure IP communication between containers. Calico implements a vRouter in each compute node that uses the kernel forwarding engine. Routes are propagated using BGP.

Used to **manage isolated networks** for Virtual Clusters

HTCondor: a batch scheduler widely used in the scientific community.

Used to **provide a familiar user experience** for batch-like use cases, and also to complement Docker's resource capping features.


HTC
HTC

Workflow

- **The user packages her full application in a Docker image**
  - industry-standard and simple, plenty of off-the-shelf base images and examples
  - The container can be run locally for testing
- **The image is pushed to OCCAM private registry**
  - OCCAM provides also a fully functional GitLab instance for CI and more, if needed
  - Only images from the private registry can be run on the system
  - Also, provides access restriction for confidential sofware unsuitable for DockerHub
- **Containers are run on OCCAM nodes**
  - Either by hand using provided occam-run CLI for simpler use cases…
  - …or by Mesosphere Marathon for complex deployments or automation
  - Normal non-admin users don't use Docker directly
- **The user can now access her private cluster**
  - one-off containers can be run exactly like batch jobs, exchange data via shared FS
  - Multi-node clusters provide an ssh service from the access node

Virtual Workstation

occam-run [-n nodename] [-i] [-x] IMAGE_NAME [CMD] [ARGS]

Returns an ID that can be used to inspect or kill the running container

Virtual Workstation

occam-run [-n nodename] [-i] [-x] IMAGE_NAME [CMD] [ARGS]

Image needs to be in OCCAM private registry

Virtual Workstation

occam-run [-n nodename] [-i] [-x] IMAGE_NAME [CMD] [ARGS]

Supports interactive containers and even X11

Virtual Workstation

occam-run [-n nodename] [-i] [-x] IMAGE_NAME [CMD] [ARGS]

Uses ssh to send the information to the nodes.
Very simple approach: on execution nodes, normal users' ssh login is replaced by ForceCommand to a script running Docker and starting the container in unprivileged mode.

udocker is an INDIGO tool to run simple containers in userspace: https://github.com/indigo-dc/udocker

Pros:

- Can be run inside a system-managed Docker container
- Less intrusive in node configuration
- Security model more obvious

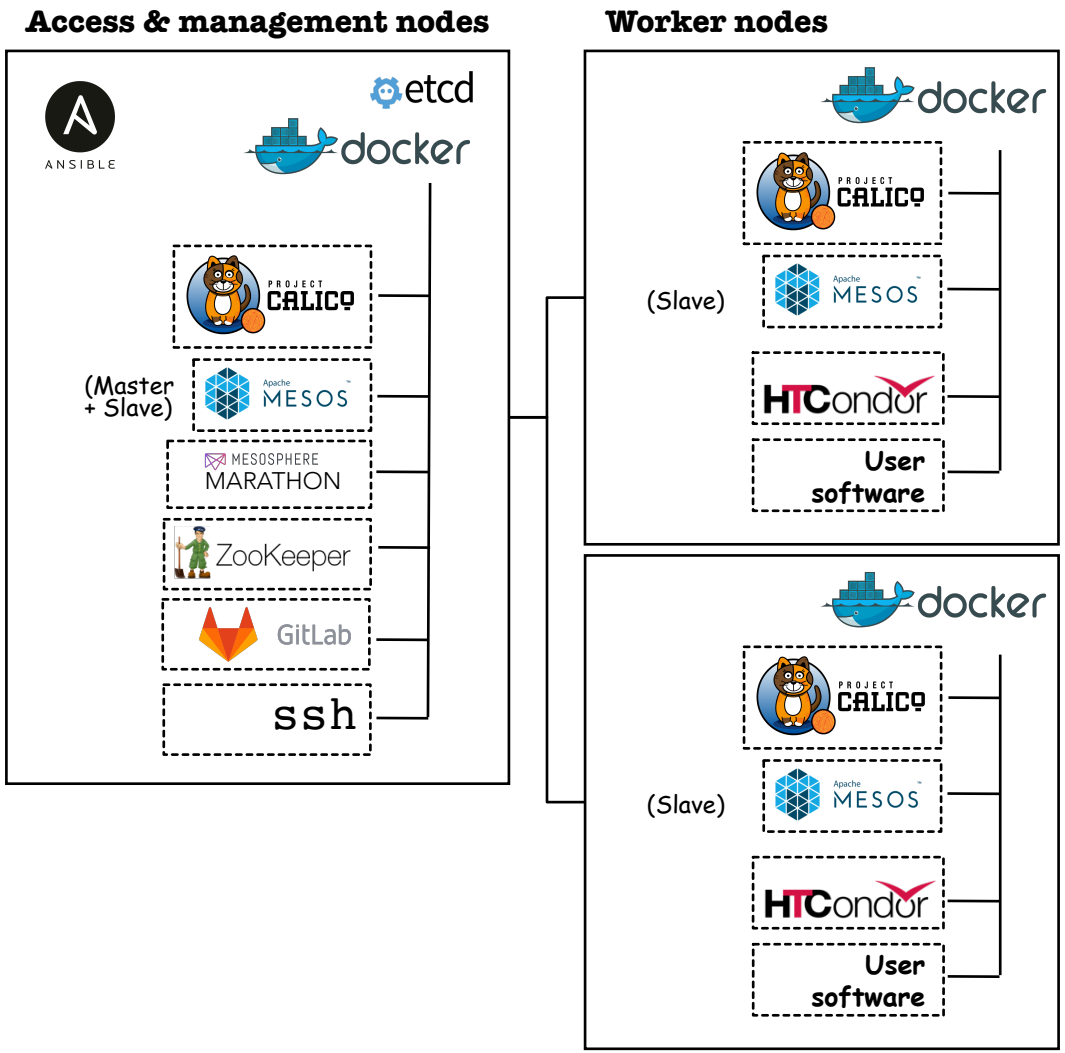
Cons:

- Application-dependent performance penalty due to proot (not in last version, to be tested)
- Recently-developed tool, not very widely used
- One more piece to maintain…

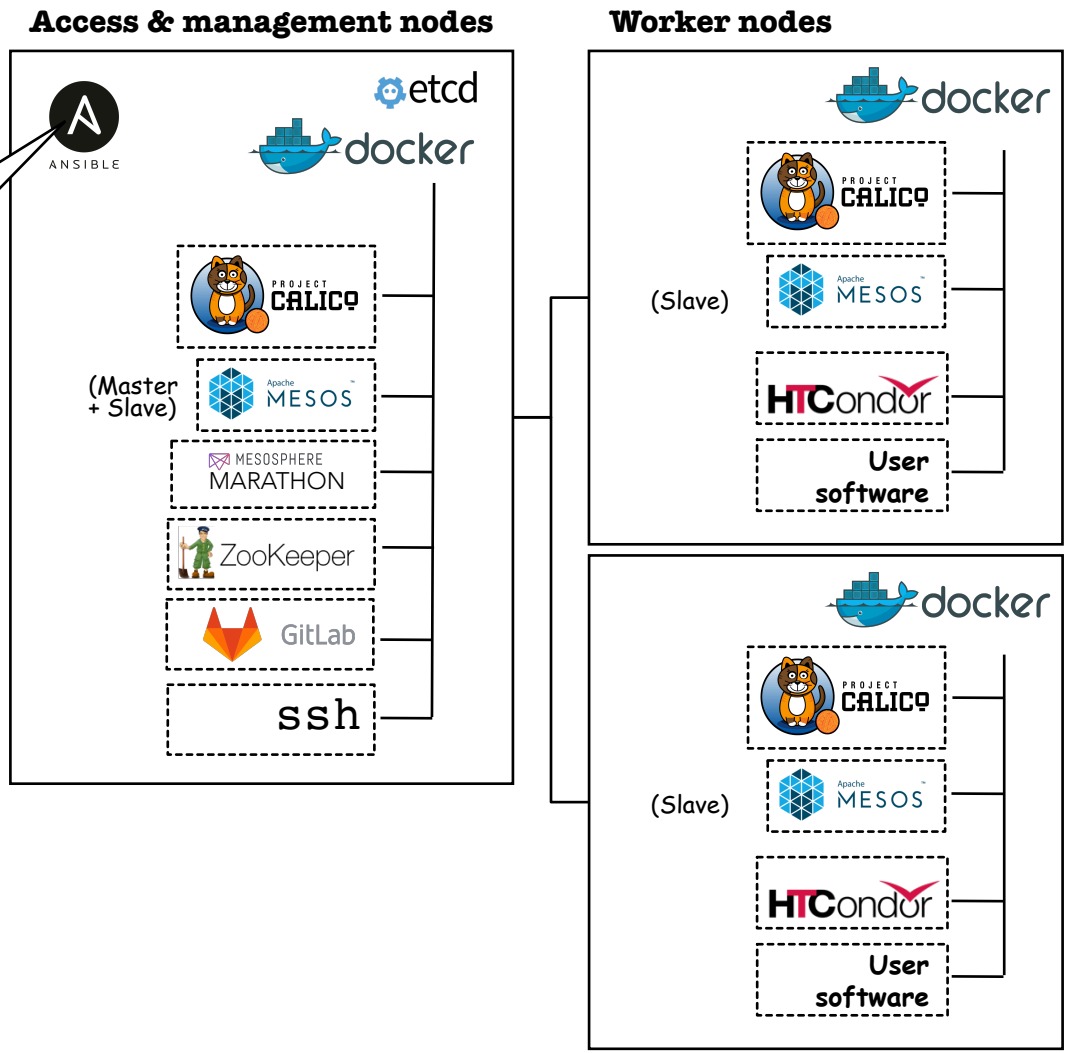
Virtual Farm

- Tools developed by INDIGO-DataCloud in the “Batch system as a service” activity
  - Provide researchers with a familiar computational framework…
  - …but using modern paradigms…
  - …and reducing administrative burden, both at infrastructure and application level.
  - The model is “one isolated virtual farm with several inner users per application”

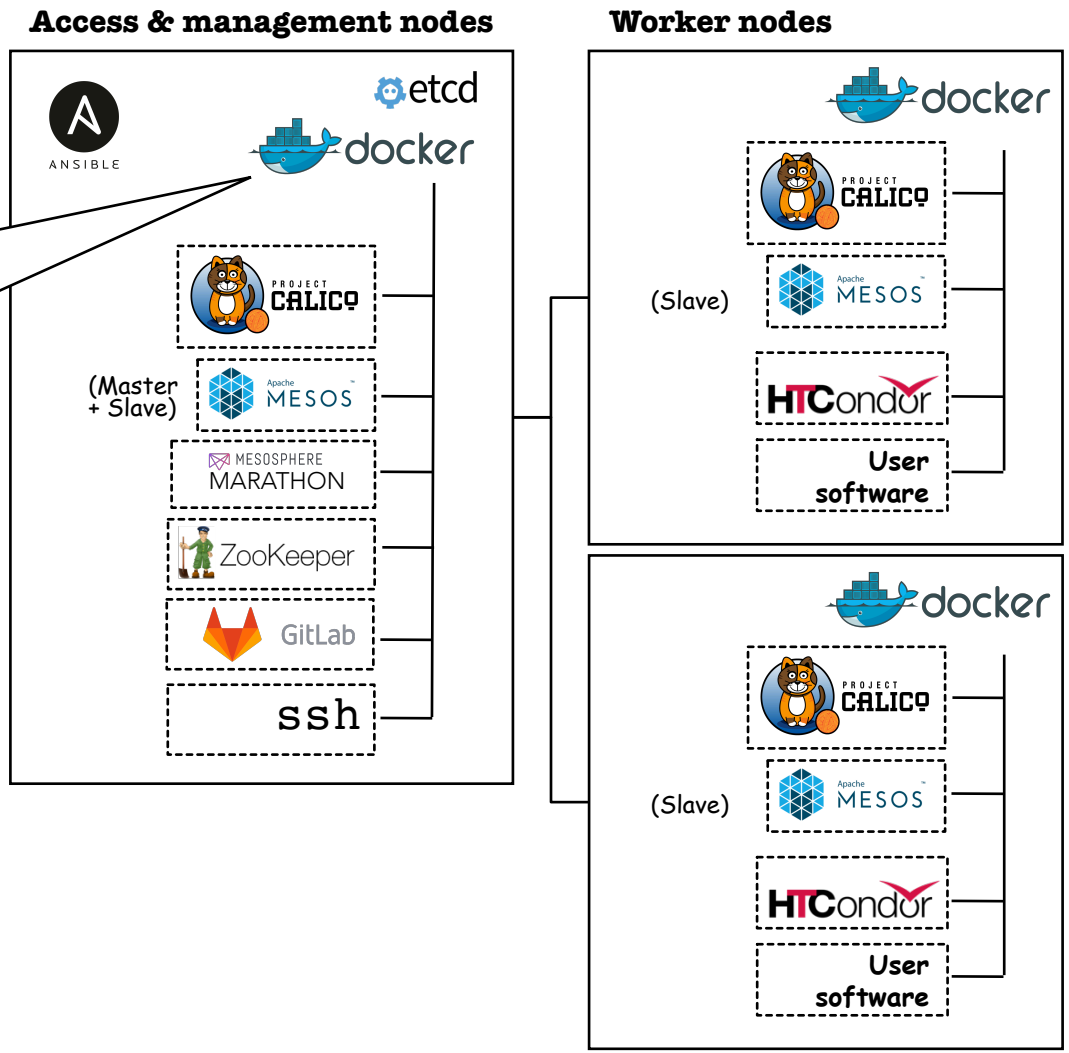
INDIGO - DataCloud


Access & management nodes
Worker nodes
ANSIBLE
etcd
docker
PROJECT CALICO
(Master + Slave)
Apache MESOS
MESOSPHERE MARATHON
ZooKeeper
GitLab
ssh
docker
PROJECT CALICO
(Slave)
Apache MESOS
HTCondor
User software
docker
PROJECT CALICO
(Slave)
Apache MESOS
HTCondor
User software

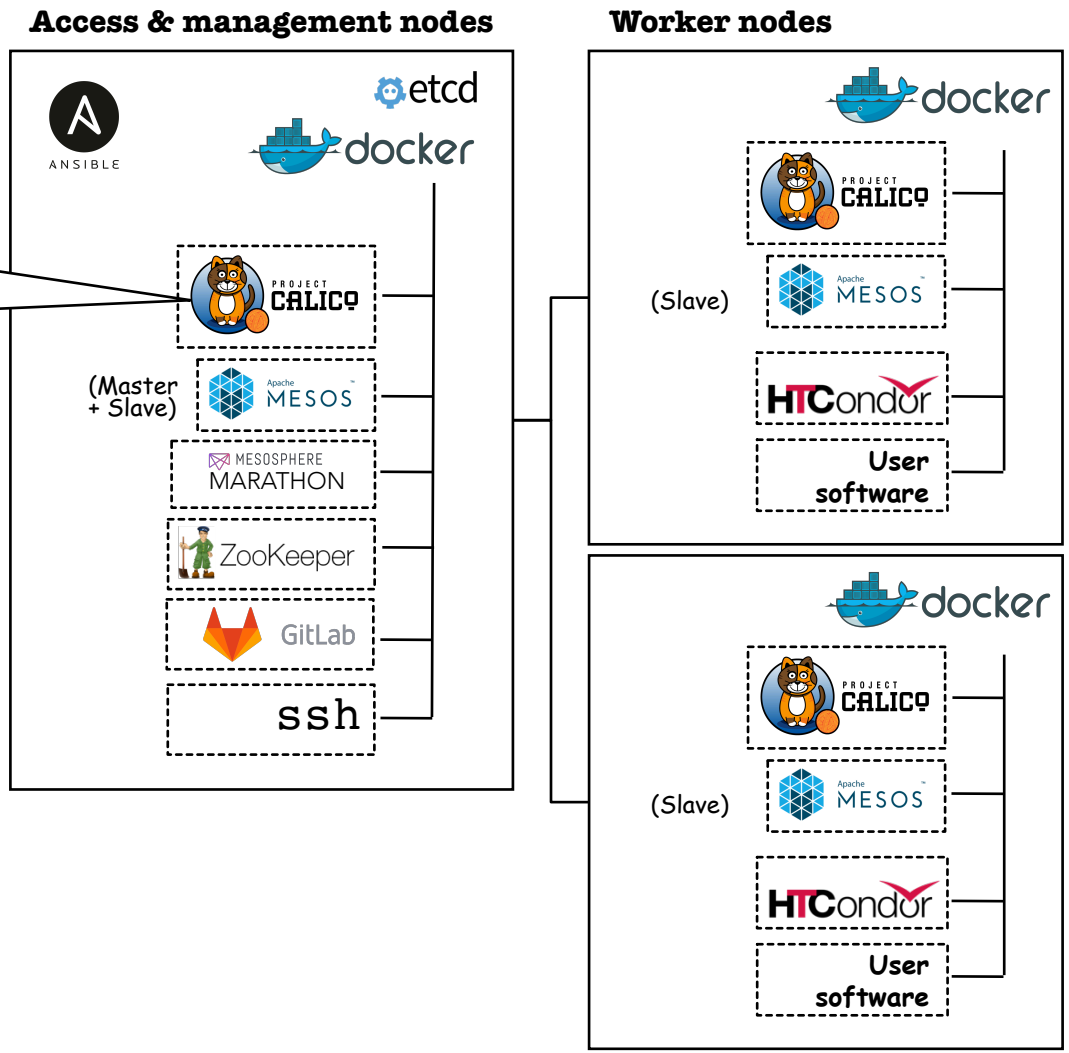
Ansible is used throughout the system for configuration management, using INDIGO-developed roles and playbooks



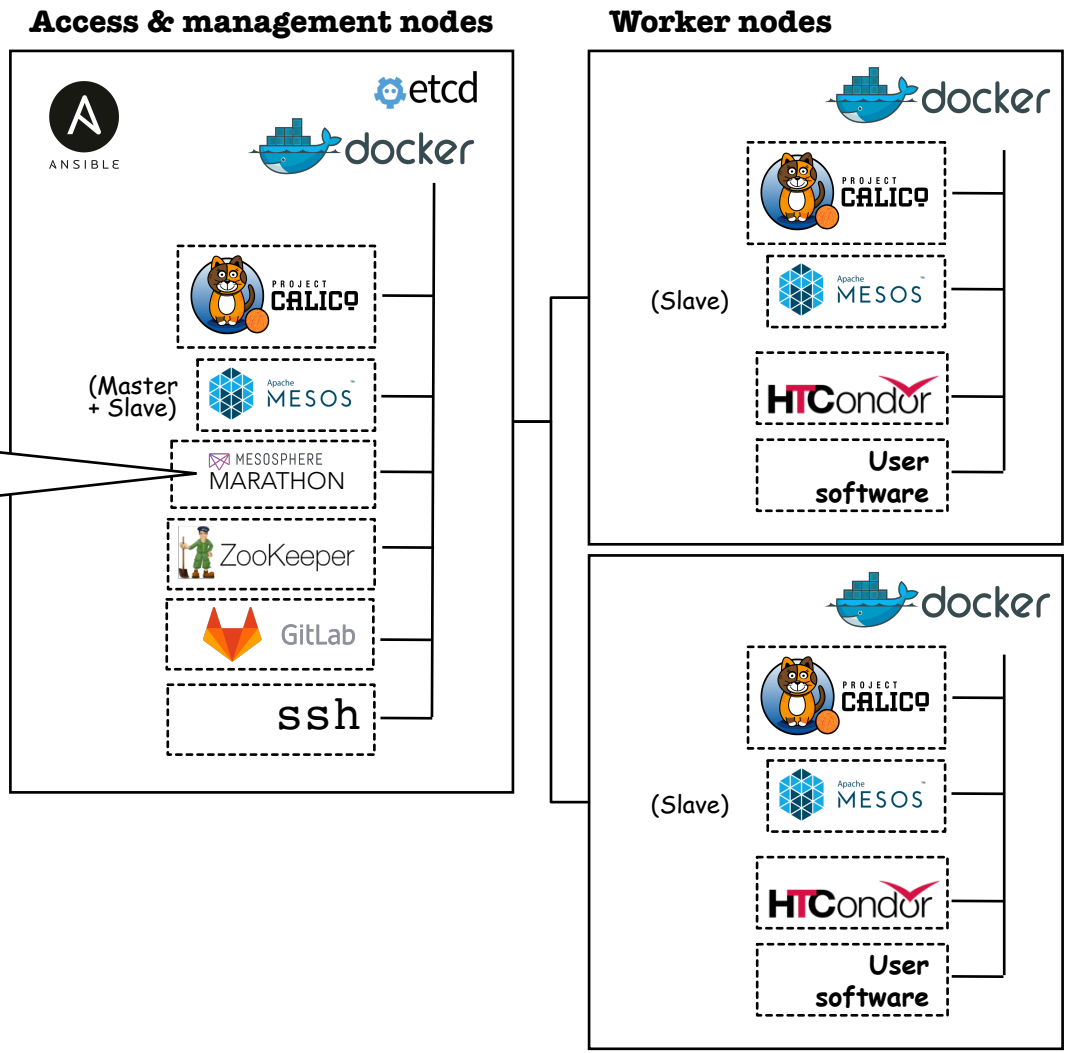
Docker is used ubiquitously to run both user software and middleware

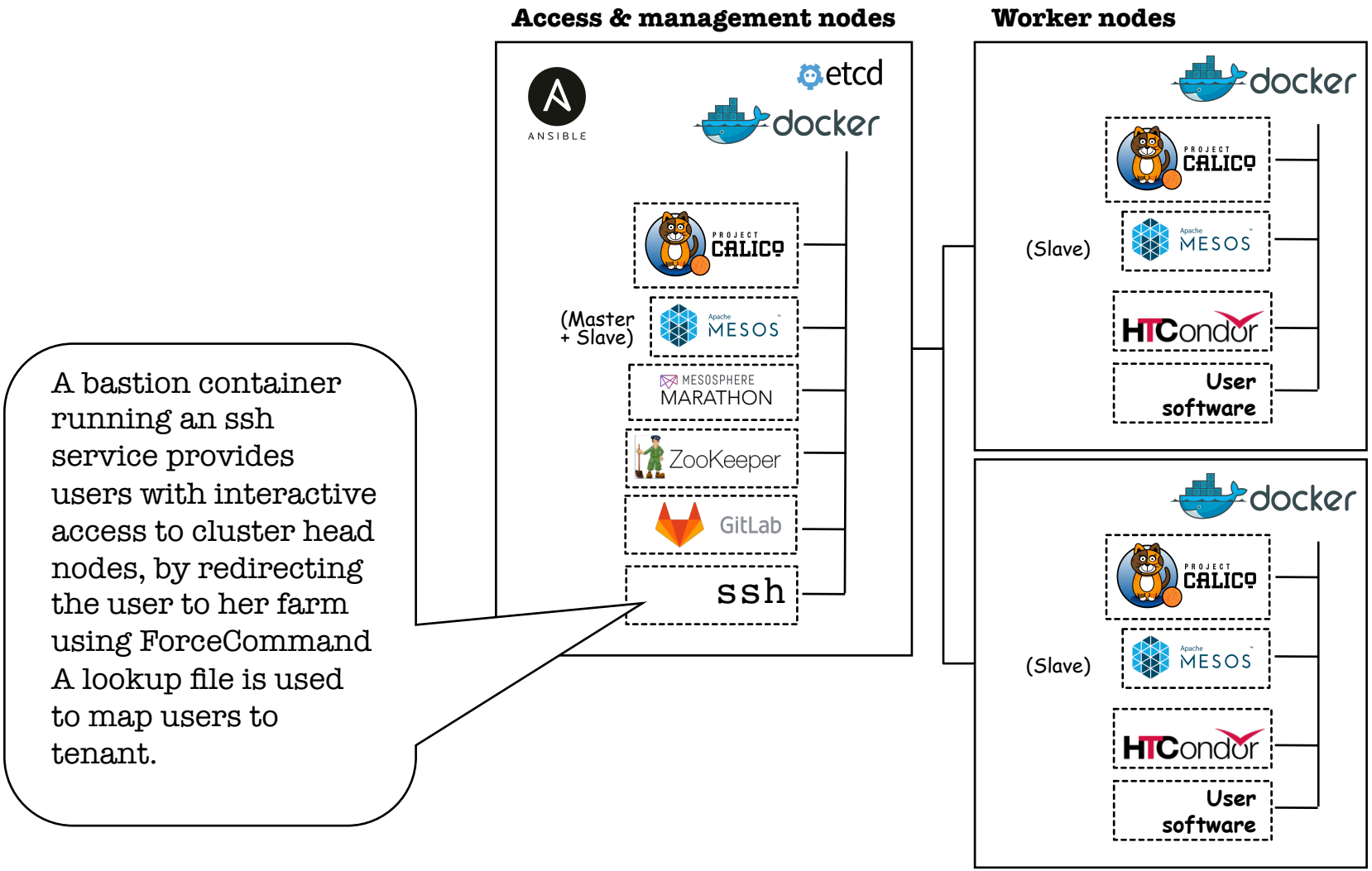


Calico manages isolated networks for each virtual farm (see also next slide)



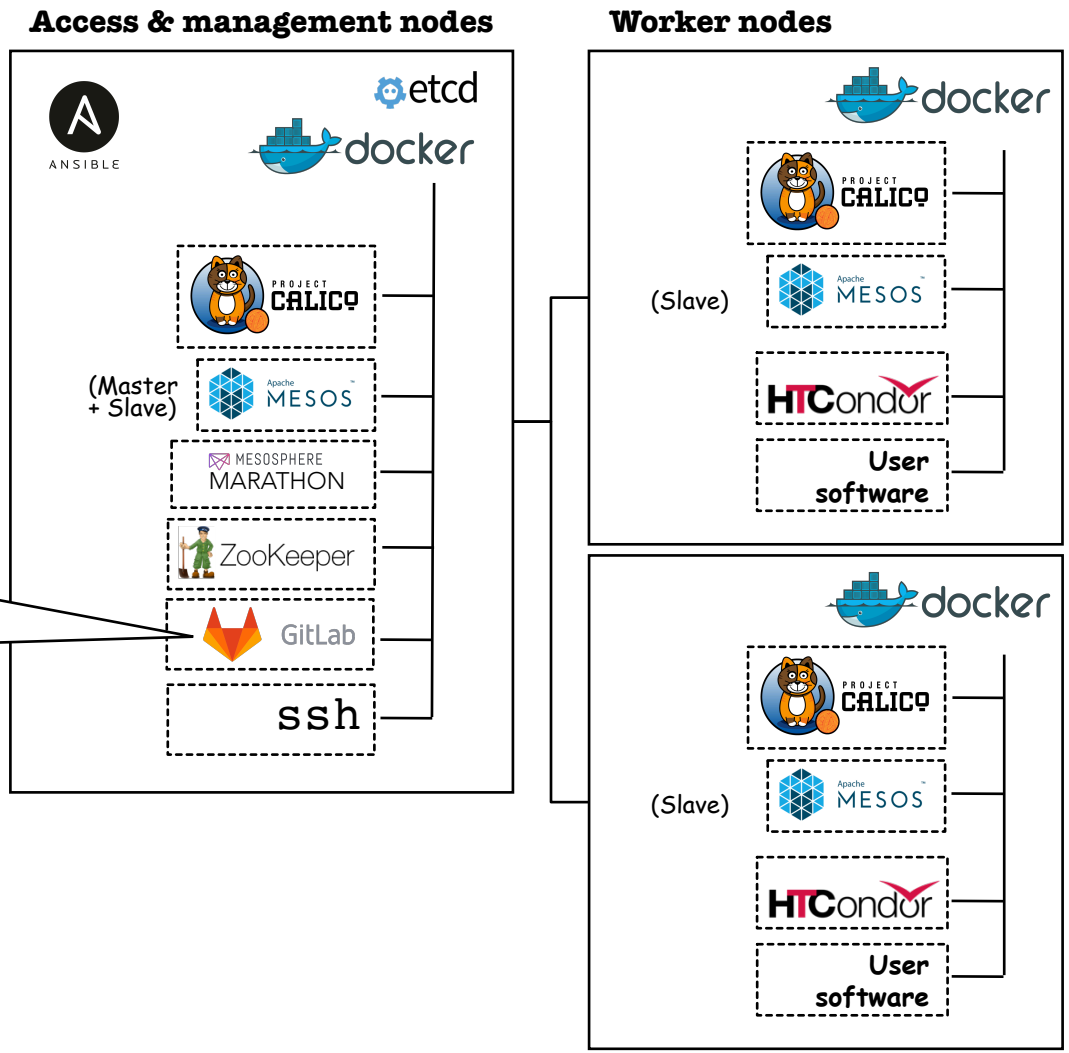
Marathon schedules and monitors user- or system- defined "executor" containers on worker nodes

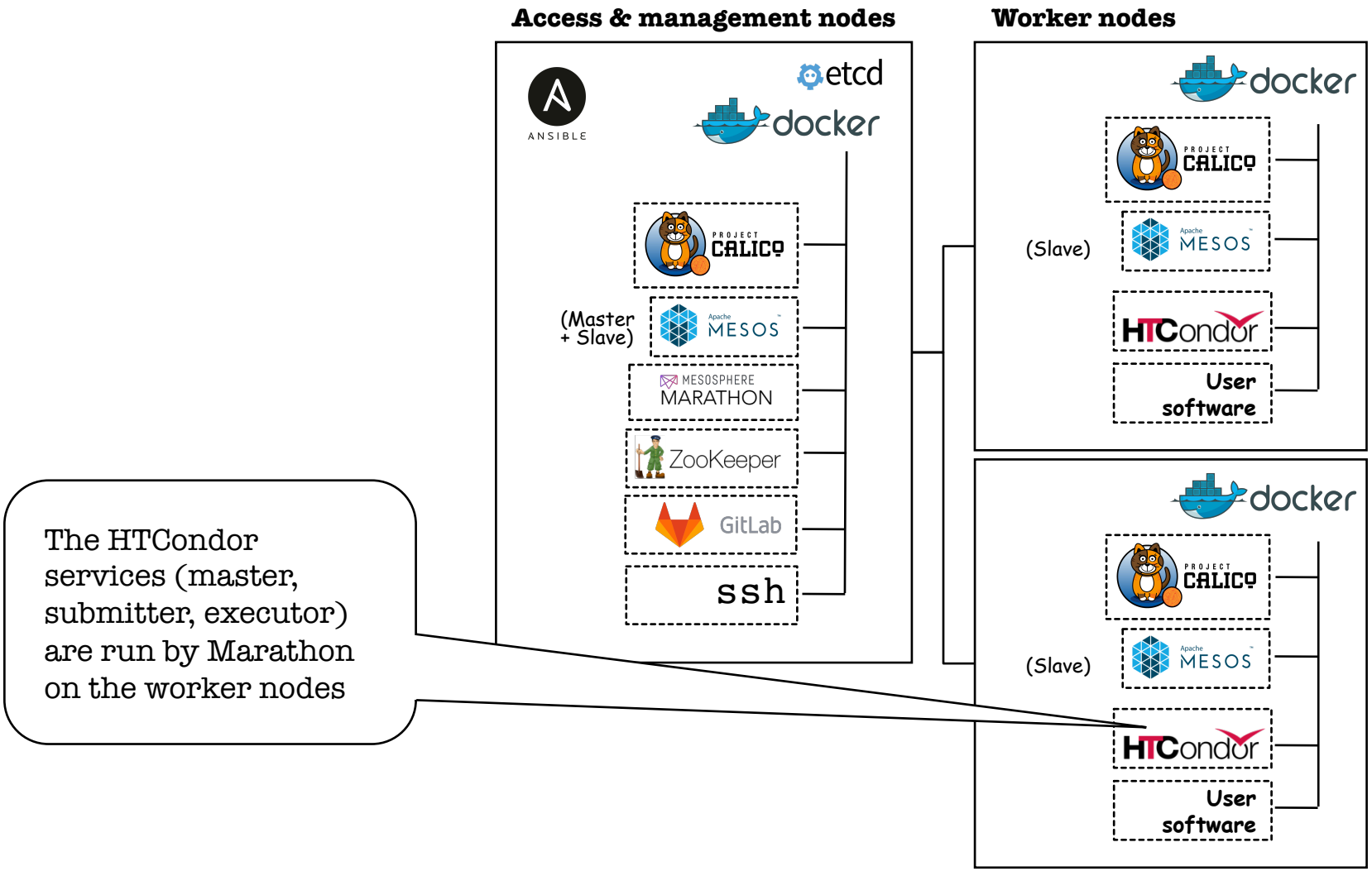




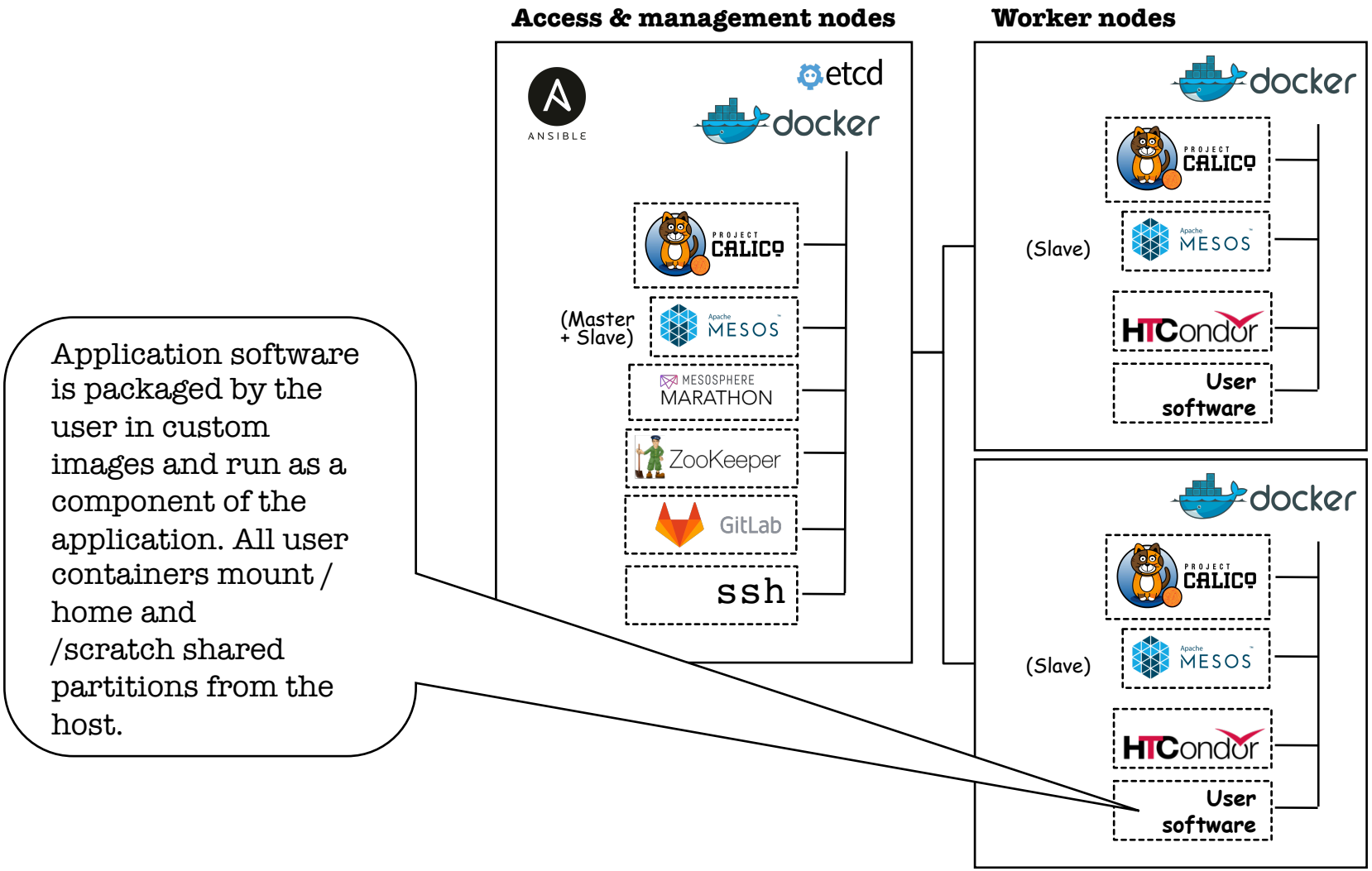
A bastion container running an ssh service provides users with interactive access to cluster head nodes, by redirecting the user to her farm using ForceCommand
A lookup file is used to map users to tenant.

GitLab is the access portal, providing user management, private image registry, continuous integration,...

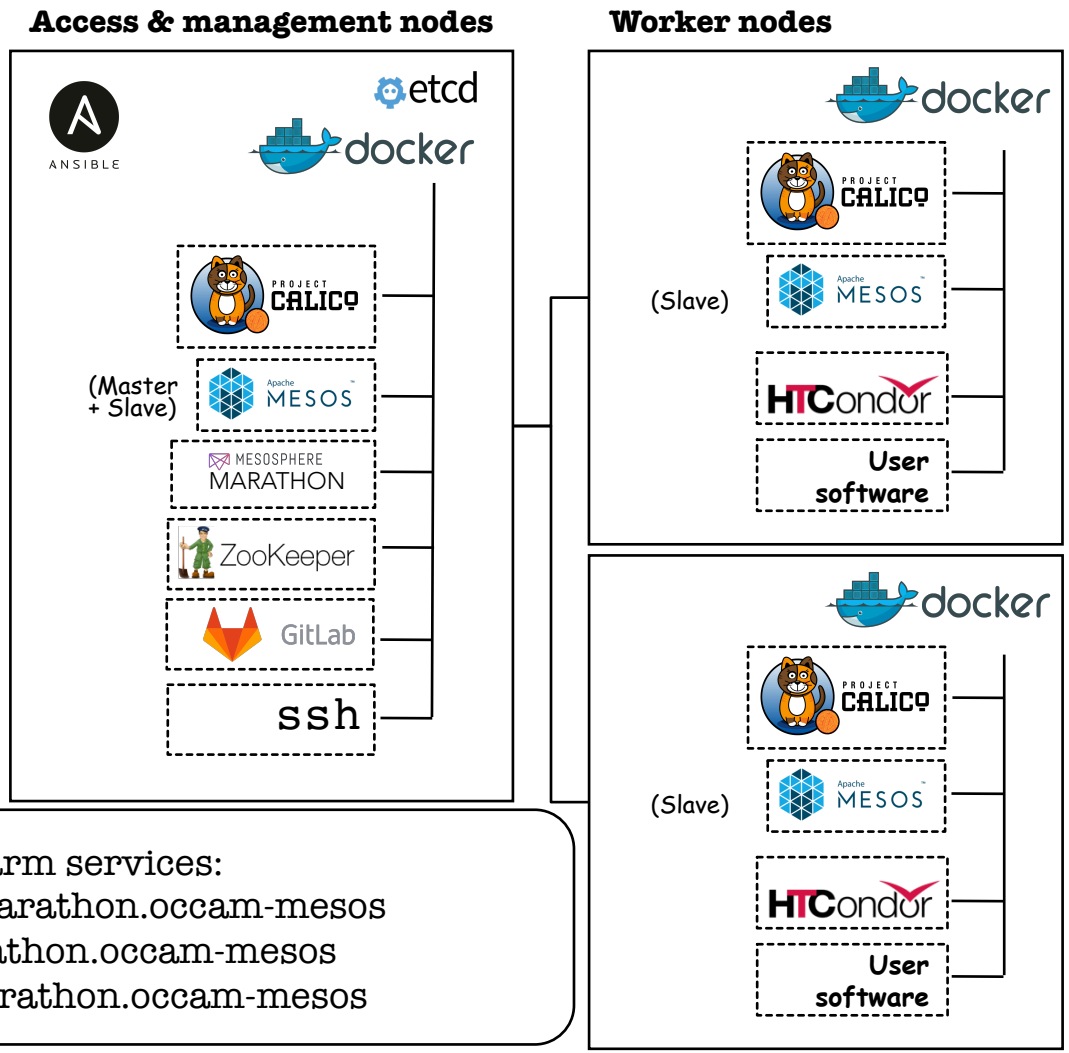




The HTCondor services (master, submitter, executor) are run by Marathon on the worker nodes



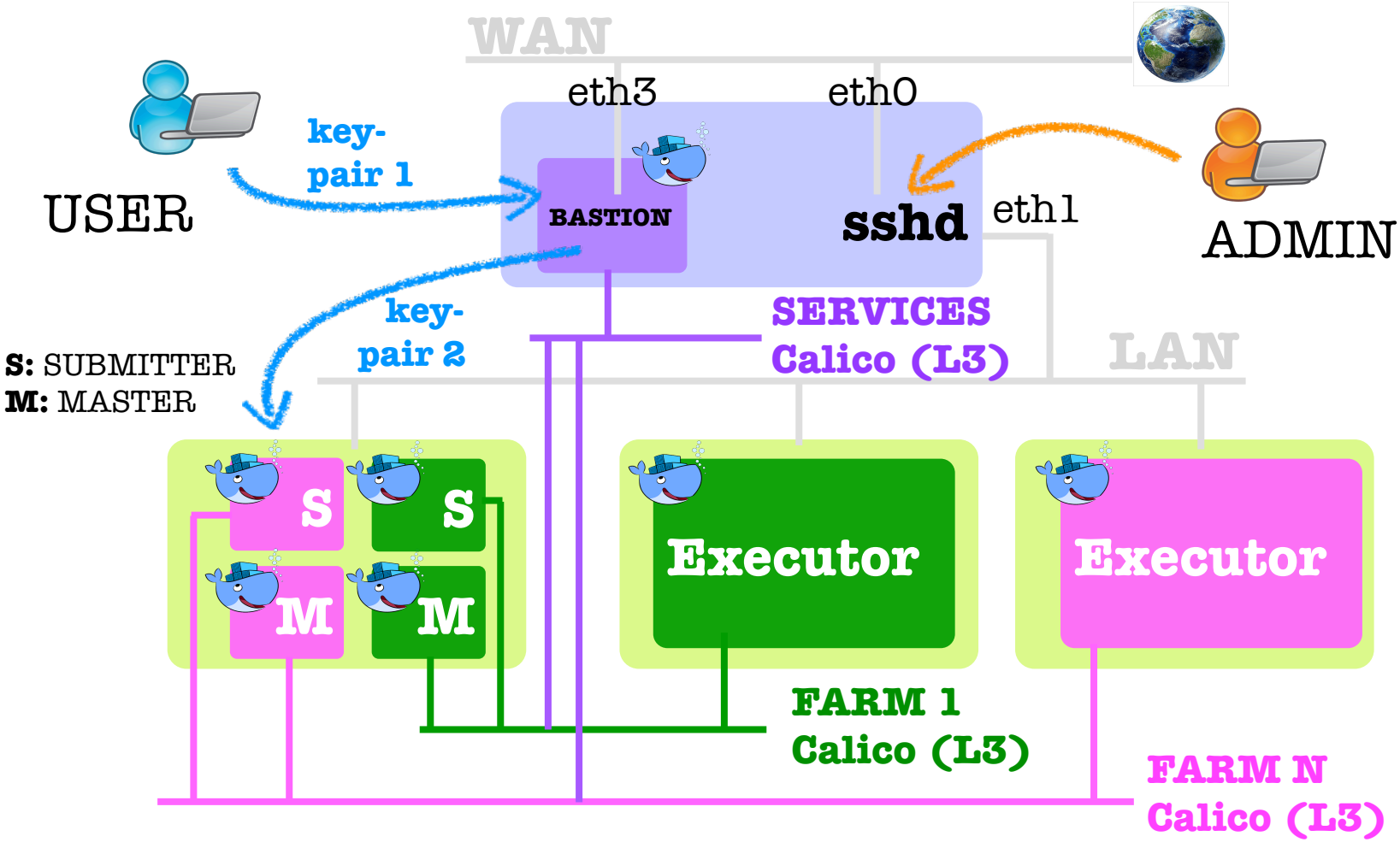
Application software is packaged by the user in custom images and run as a component of the application. All user containers mount /home and /scratch shared partitions from the host.



Mesos-DNS is used to name farm services:

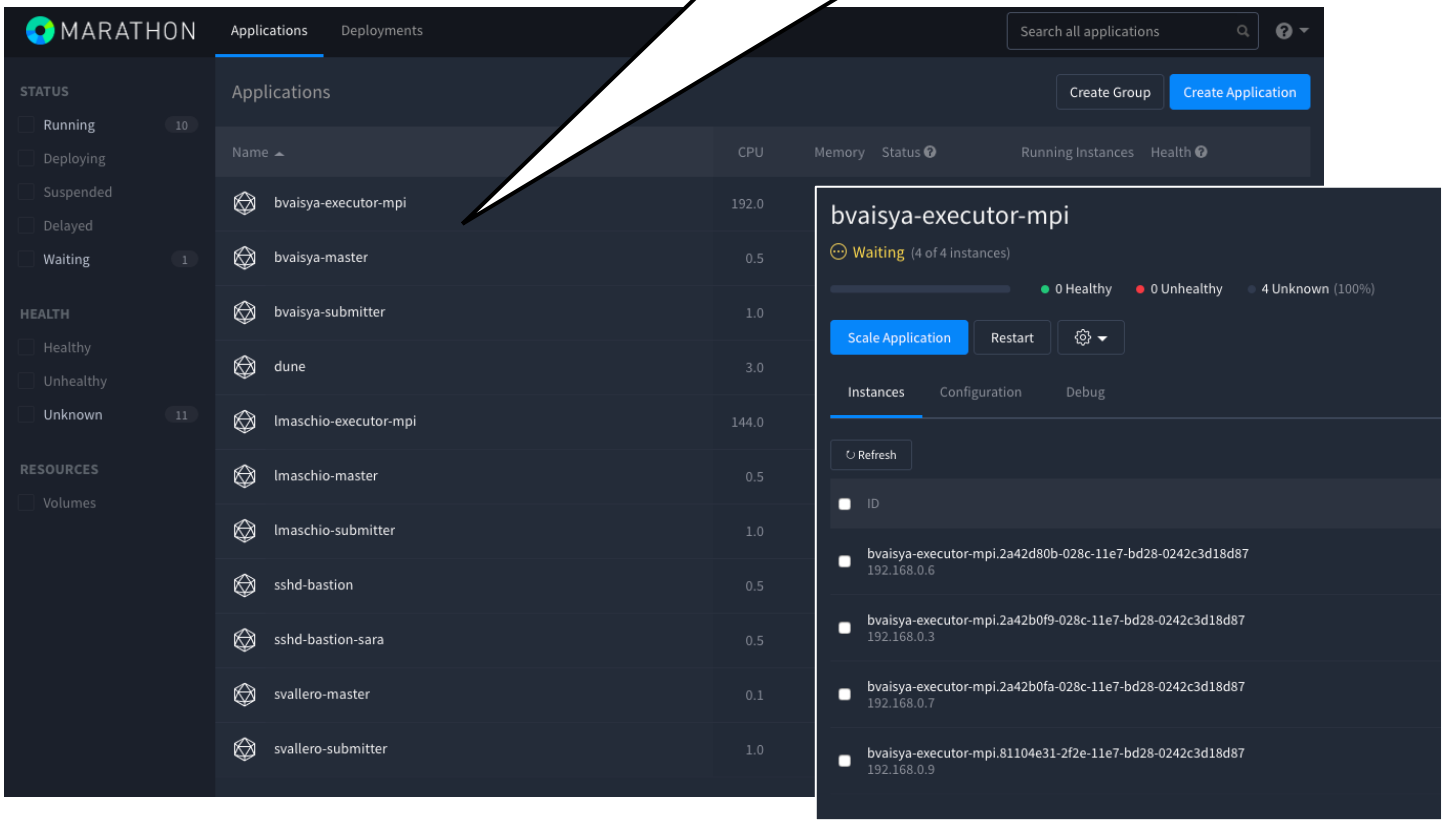
<tenant>-submitter.marathon.occam-mesos
<tenant>-master.marathon.occam-mesos
<tenant>-executor.marathon.occam-mesos

Networks


WAN
eth3
eth0
USER
key-
pair 1
BASTION
sshd
eth1
ADMIN
key-
pair 2
SERVICES
Calico (L3)
LAN
S: SUBMITTER
M: MASTER
S
S
M
M
Executor
Executor
FARM 1
Calico (L3)
FARM N
Calico (L3)

Marathon

HTCondor component containers (Master, Submitter and Executor) deployed as Marathon applications



Notes

- Integration of user containers in a multi-node virtual farm still needs manual intervention from admins
  - Will be automated at some point
- HTCondor proved to be an overkill
  - The “embedded” batch system has only one queue
  - htcondor HPC features are not widely used, so they are not as streamlined as other
- We provide custom base images for special uses or ease of integration
  - ssh executor
  - GPU support
  - …
- We are still looking for a tool to manage the Pipeline use case
  - Galaxy?

More notes

- This model is ambitious
  - Without it OCCAM is just another smallish HPC facility
  - Took a while to start, but activity is gaining momentum (3 VF activities, o(20) VW users,...)
- This approach requires users to somehow change their workflow
  - The feedback is insofar mostly positive
  - And generally we encourage users to adopt modern technologies and DevOps ideas
- Generally users learn quickly to use Docker and appreciate the tool
  - They don't need the more subtle features, just to write a dockerfile and build
  - The learning curve is smooth (Docker has very good docs!)

Thanks!

The OCCAM cluster and the Centro di Competenza sul Calcolo Scientifico of the University of Torino were funded though a contribution by Compagnia di San Paolo

The architecture

32 "Light" nodes

- CPU - 2x Intel® Xeon® Processor E5-2680 v3, 12 core 2.5Ghz
- RAM - 128GB/2133 (8 x 16 Gb)
- DISK - SSD 400GB SATA 1.8 inch.
- NET - IB 56Gb + 2x10Gb
- High density form factor (4 nodes x RU)

4 "Fat" nodes

- CPU - 4x Intel® Xeon® Processor E7-4830 v3 12 core/2.1Ghz
- RAM - 768GB/1666MHz (48 x 16Gb) DDR4
- DISK - 1 SSD 800GB + 1 HDD 2TB 7200rpm
- NET - IB 56Gb + 2x10Gb

4 "GPU" nodes

- CPU - 2x Intel® Xeon® Processor E5-2680 v3, 12 core 2.1Ghz
- RAM - 128GB/2133 (8 x 16Gb) DDR4
- DISK - 1 x SSD 800GB sas 6 Gbps 2.5"
- NET - IB 56Gb + 2x10Gb
- GPU - 2 x NVIDIA K40 su PCI-E Gen3 x16

High-performance "Scratch" storage

- DISK TYPE - HDD da 4 TB SAS 7200 rpm
- CAPACITY - 320 TB RAW e 256 TB usable
- NET - 2 x IB 56Gb FDR + 2 x 10Gb
- FILESYSTEM - Lustre Parallel Filesystem

"Archival" (non-custodial) storage

- DISK TYPE - 180 x 6 TB a 7200 rpm SAS 6Gbps
- CAPACITY - 1080 TB raw (768 TB usable)
- NET - 2 x IB 56Gb + 4 x 10GbE
- FILESYSTEM - NFS export
- Dynamic Disk Pools equivalent to RAID 6

Networking:

- InfiniBand layer - 56 Gbps "Fat Tree"
- 10GBPS Ethernet - 10 Gbps flat
- 1GBPS Ethernet for monitoring and management