INF9380

Cloud computing intro

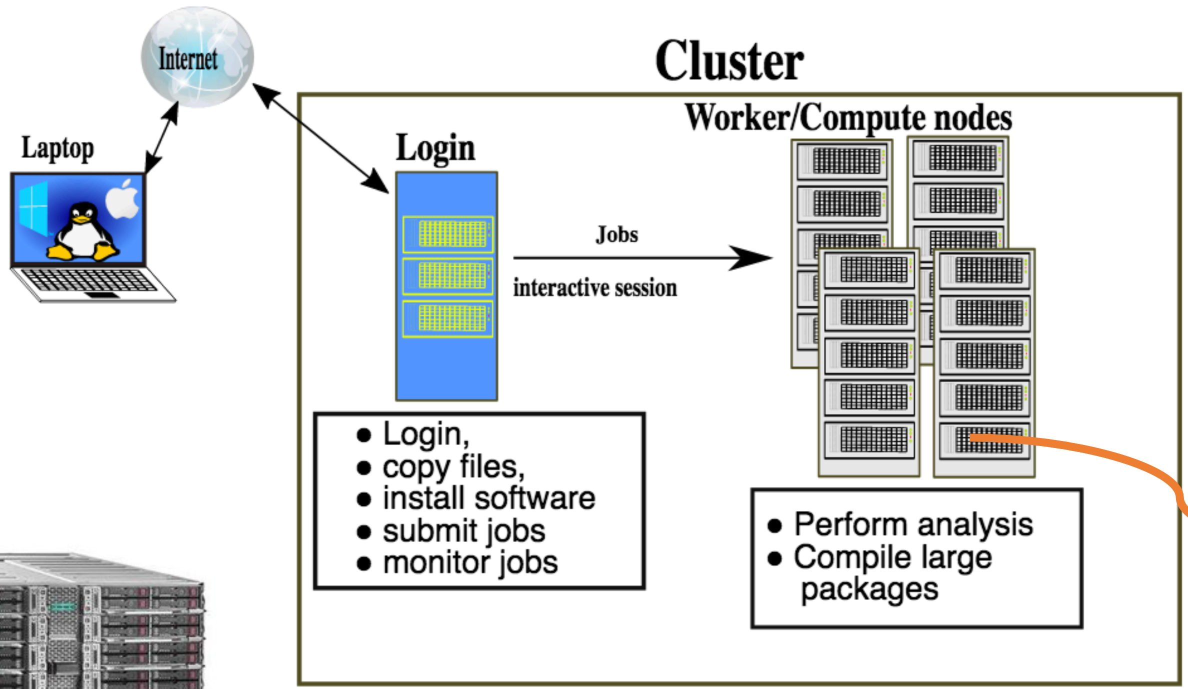
Outline of cloud session

1. Cloud, HPC, Scheduling systems - rough overview
2. Create a cluster on the cloud – Elasticluster tutorial
   - Intro to Elasticluster
   - Set up and submit a job to your own cluster - tutorial
3. Distributed computing – computing grids – example with Worldwide LHC computing grid (WLCG)
   - Introduction to the WLCG
   - Send a job to the grid

Part 1.
Cloud, HPC, Scheduling systems rough overview

"Traditional" computing cluster resource versus cloud resource

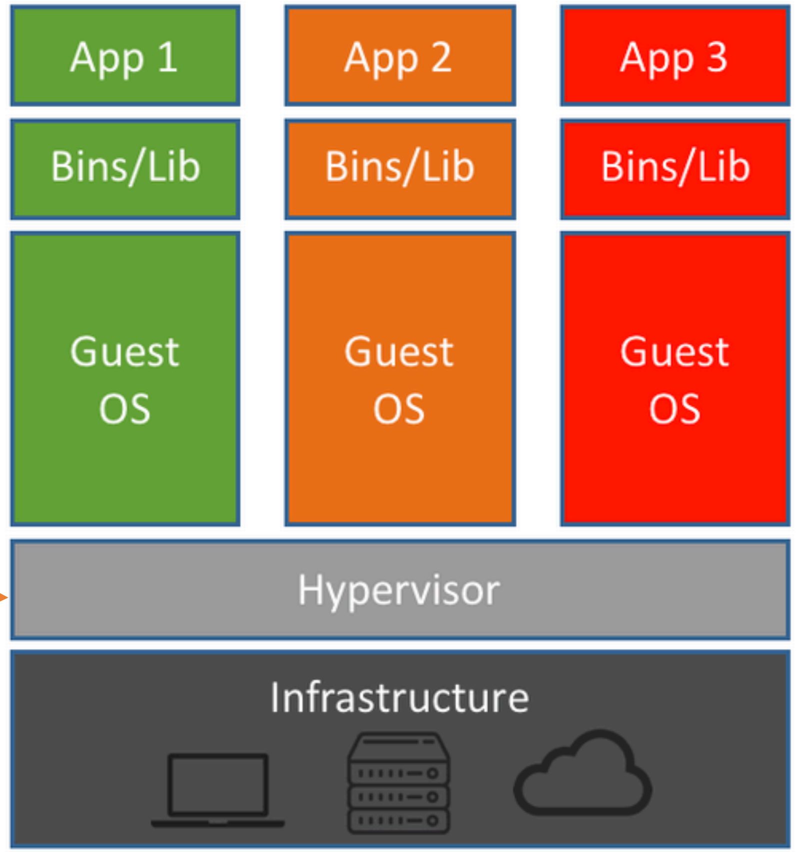
Traditional cluster



Node: One computer with X number of processors, each with Y number of cores

Example: 1 node with 2 processors, each with 20 cores per processor. Total cores: 40

Cloud Virtualisation



Machine Virtualization

“Traditional” computing cluster resource

Good, but non-flexible

- All machines have the same CPU chip architecture, let’s say Intel or AMD processors
- High performance processors, not the regular standard processors (usually)
- All machines have the same OS, e.g. Red Hat 8
- Some speciality nodes: big-mem, GPU’s
- Fast interconnect between nodes
- Access to common shared file system
- Queue system, some times with a long waiting time

Queueing system (SLURM):
“Please wait in line.
The big-mem nodes you are requesting
are busy, and will be so for another 2 weeks.
If you hold the line you will move forward
In the queue. “

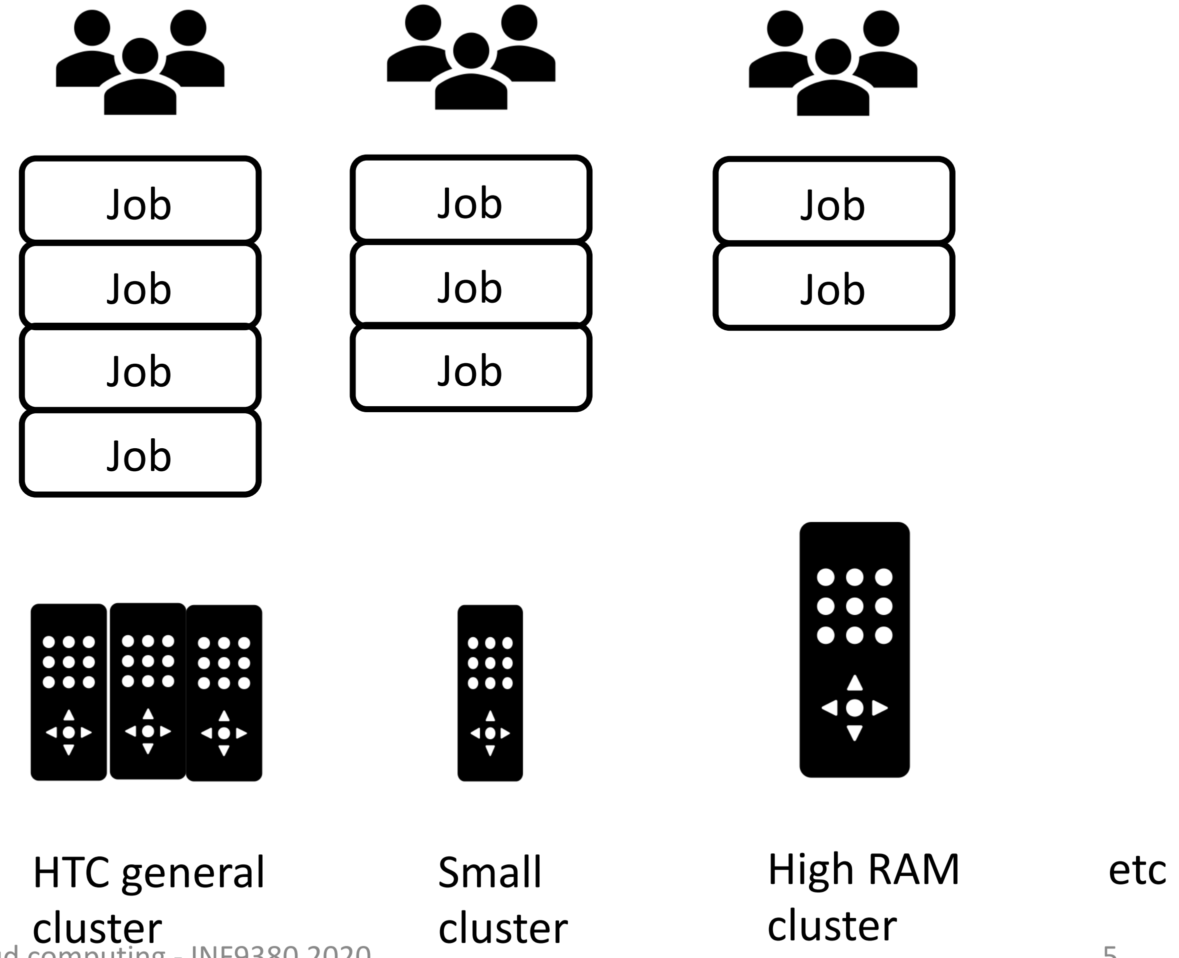
User:

☹ ☹ ☹

Fast and powerful system,
but waiting time can be long

HPC on cloud

- Virtualisation is used to provision servers
- Dynamic set of servers – i.e. virtual servers are fired up as needed, and destroyed when not needed anymore
- You can pick and choose the type of (virtual) server you want: size (number of virtual cpu's, size of RAM, size of disk etc etc)
- Can set up as many queues as you want or as many clusters as you want – you create the cluster that suits your needs
- You do not have to wait in line!
- Flexible!!
- Not necessarily high performance hardware – can be "off-the-shelf" processors
- Not necessarily very good interconnect between nodes (no infiniband)



When would you use HPC on cloud?

- Example use-case:
  - You would like to run some application that requires more cores than you have available (general use-case for cluster computing) and that can benefit from more cores i.e. be parallelized
  - You have access to a cloud resource
  - You do not have access to an HPC resource, or you do not want to use your quota there, or don’t want to wait in line
  - …

Some cloud platforms/providers

- Google Cloud Platform
- Amazon Web Services (AWS)
- Microsoft Azure
- Oracle Cloud
- ...https://www.guru99.com/cloud-computing-service-provider.html

Public - commercial

Google Cloud Platform

Microsoft Azure

amazon web services™

ORACLE CLOUD

- Openstack - free open standard
  - NREC: *Norwegian* Research and Education Cloud
  - This is what we will be working in

openstack®

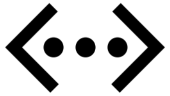
Workflow - clouds

- Launch
  - Create the virtual machine with a base OS

- Configure
  - Install necessary software like e.g. database software, web application software, scientific software etc etc

Automation tools

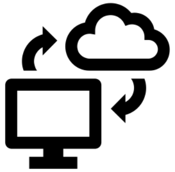
- Connect
  - Remotely connect to your instance via ssh



Terminology and workflow - clouds

- Region
  - When you create virtual machines you do that in a geographical region – e.g. North Europe, Oslo, Bergen etc
  - Your virtual machines will then be created on host machines in a data center belonging to that region
  - Can be performance issues and/or political reasons for choosing a certain region

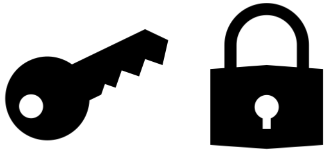
- Instance
  - A virtual machine



- Images
  - What OS image to install on your instance

- Flavor
  - The specifications of the virtual machine:
    - Number of virtual CPU's (vCPU)
    - Amount of RAM
    - Amount of disk
    - Processor type (AMD/Intel/ARM)

- Key pairs
  - To connect to your instances you use an ssh private-public key pair

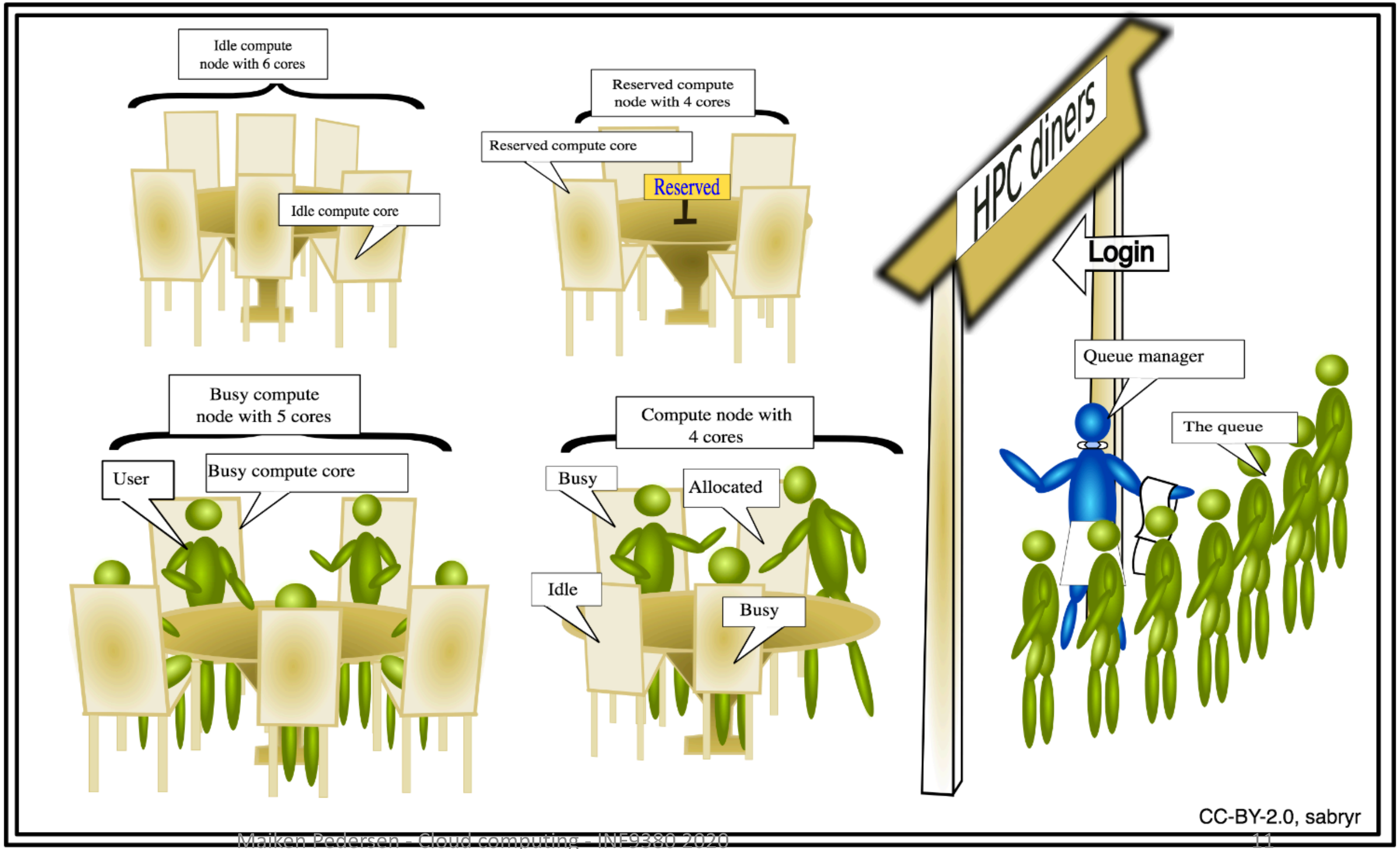


- Network security group
  - Set of network rules

Working with HPCs or other clusters with a scheduling system

Computing on a cluster – scheduling systems

- To handle the compute jobs that are submitted by the users, the cluster has a queuing system installed
- Examples: SLURM, HTCondor, PBS, Torque…
- Job is scheduled based on: Priorities, special requests (node with large RAM), cpu-hours left in project, etc etc



https://hpc-carpentry.github.io/hpc-intro/13-scheduler/index.html

```
[centos@frontend001 ~]$ squeue
     JOBID PARTITION     NAME     USER ST       TIME  NODES   CPUS NODELIST(REASON)
    219365      main data16_1     grid PD       0:00      1      7      (Resources)
    219366      main data16_1     grid PD       0:00      1      7       (Priority)
    219367      main data16_1     grid PD       0:00      1      7       (Priority)
    219370      main data16_1     grid PD       0:00      1      7       (Priority)
    219369      main data16_1     grid PD       0:00      1      7       (Priority)
    219375      main data16_1     grid PD       0:00      1      7       (Priority)
    219377      main data16_1     grid PD       0:00      1      7       (Priority)
    219349      main mc16_13T     grid PD       0:00      1      7       (Priority)
    219351      main mc16_13T     grid PD       0:00      1      7       (Priority)
    219371      main mc16_13T     grid PD       0:00      1      7       (Priority)
    219372      main mc16_13T     grid PD       0:00      1      7       (Priority)
    219374      main mc16_13T     grid PD       0:00      1      7       (Priority)
    219381      main data16_1     grid PD       0:00      1      7       (Priority)
    219383      main data16_1     grid PD       0:00      1      7       (Priority)
    219384      main data16_1     grid PD       0:00      1      7       (Priority)
    219386      main data16_1     grid PD       0:00      1      7       (Priority)
    219389      main data16_1     grid PD       0:00      1      7       (Priority)
    219481      main mc16_13T     grid PD       0:00      1      1       (Priority)
    219364      main data16_1     grid  R       2:40      1      8       compute041
    219363      main data16_1     grid  R       2:50      1      8      computec023
    219362      main data16_1     grid  R       6:47      1      8       compute031
    219361      main data16_1     grid  R       8:42      1      8      computec010
    219359      main data16_1     grid  R      15:10      1      8      computeb007
    219360      main data16_1     grid  R      14:44      1      8      computec004
    219333      main mc16_13T     grid  R      19:21      1      8      computeb009
    219332      main mc16_13T     grid  R      21:53      1      8       compute039
    213618      main mc15_13T     grid  R 5-07:37:11      1      1       compute035
    219331      main mc16_13T     grid  R      33:07      1      8      computec011
    219330      main mc16_13T     grid  R      33:09      1      8       compute032
    219343      main data16_1     grid  R    1:02:21      1      8       compute036
    219354      main data16_1     grid  R      42:41      1      8       compute046
    219325      main mc16_13T     grid  R    1:02:08      1      8       compute033
    219358      main data16_1     grid  R      36:05      1      8      computec017
    219327      main mc16_13T     grid  R      38:13      1      8      computec014
```

Slurm command examples:

- squeue
- sinfo
- sbatch
- srun

```
[centos@frontend001 ~]$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
main*        up   infinite      1    mix compute035
main*        up   infinite     63  alloc compute[002,005,011,014-015,021,030-034,036-049],computeb[001-010],computec[001-028]
```

```
#!/bin/bash
#
#SBATCH --job-name=student00
#SBATCH --output=student00_slurmid_%j.out
#
#SBATCH --ntasks=1
#SBATCH --time=10:00
#SBATCH --mem-per-cpu=100

srun hostname
srun hostname -i
srun sleep 60
```

Number of parallel processes

- Submit job with sbatch <jobscript-name>
- https://slurm.schedmd.com/sbatch.html

```
#!/bin/bash

#SBATCH --ntasks=1

srun sleep 10 &
srun sleep 12 &
wait
```

```
sacct -j515058 --format=JobID,Start,End,Elapsed,NCPUS

       JobID               Start                 End    Elapsed      NCPUS
------------ ------------------- ------------------- ---------- ----------
515058       2018-12-13T20:51:44 2018-12-13T20:52:06   00:00:22          1
515058.batch 2018-12-13T20:51:44 2018-12-13T20:52:06   00:00:22          1
515058.0     2018-12-13T20:51:44 2018-12-13T20:51:56   00:00:12          1
515058.1     2018-12-13T20:51:56 2018-12-13T20:52:06   00:00:10          1
```

```
#!/bin/bash

#SBATCH --ntasks=2

srun --ntasks=1 sleep 10 &
srun --ntasks=1 sleep 12 &
wait
```

```
sacct -j 515064 --format=JobID,Start,End,Elapsed,NCPUS
JobID                    Start                 End    Elapsed      NCPUS
------------ ------------------- ------------------- ---------- ----------
515064       2018-12-13T21:34:08 2018-12-13T21:34:20   00:00:12          2
515064.batch 2018-12-13T21:34:08 2018-12-13T21:34:20   00:00:12          2
515064.0     2018-12-13T21:34:08 2018-12-13T21:34:20   00:00:12          1
515064.1     2018-12-13T21:34:08 2018-12-13T21:34:18   00:00:10          1
```