Using the general purpose computing clusters at EPFL

`scitas.epfl.ch`

September 14, 2018

Welcome

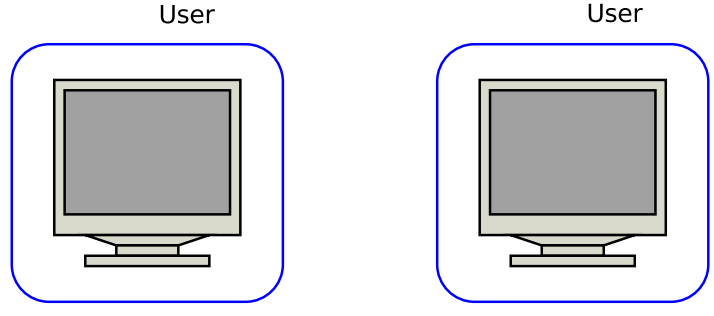
What you will learn

- What is a cluster
- What is a scheduler
- How the environment is organised
- How to run simple jobs on our clusters

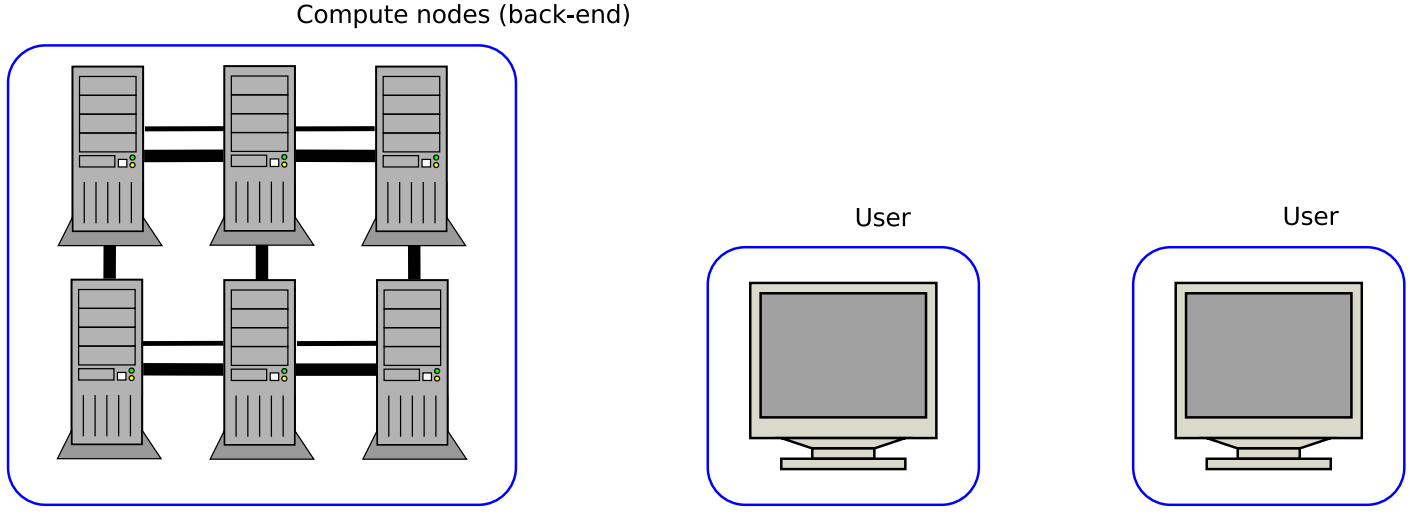
What you will not learn

- Writing and (compiling) codes
- Parallelising code

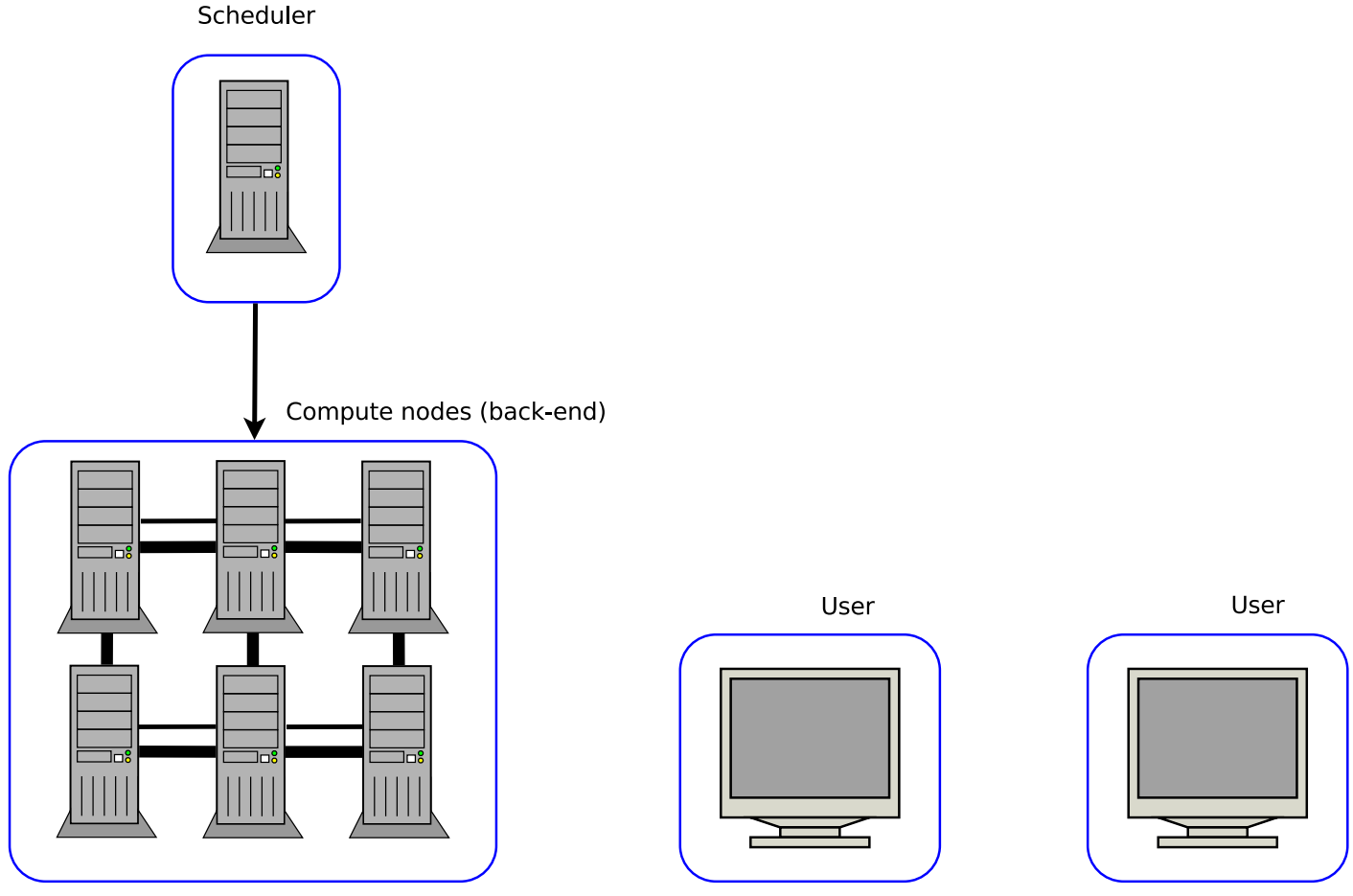
What is a cluster?



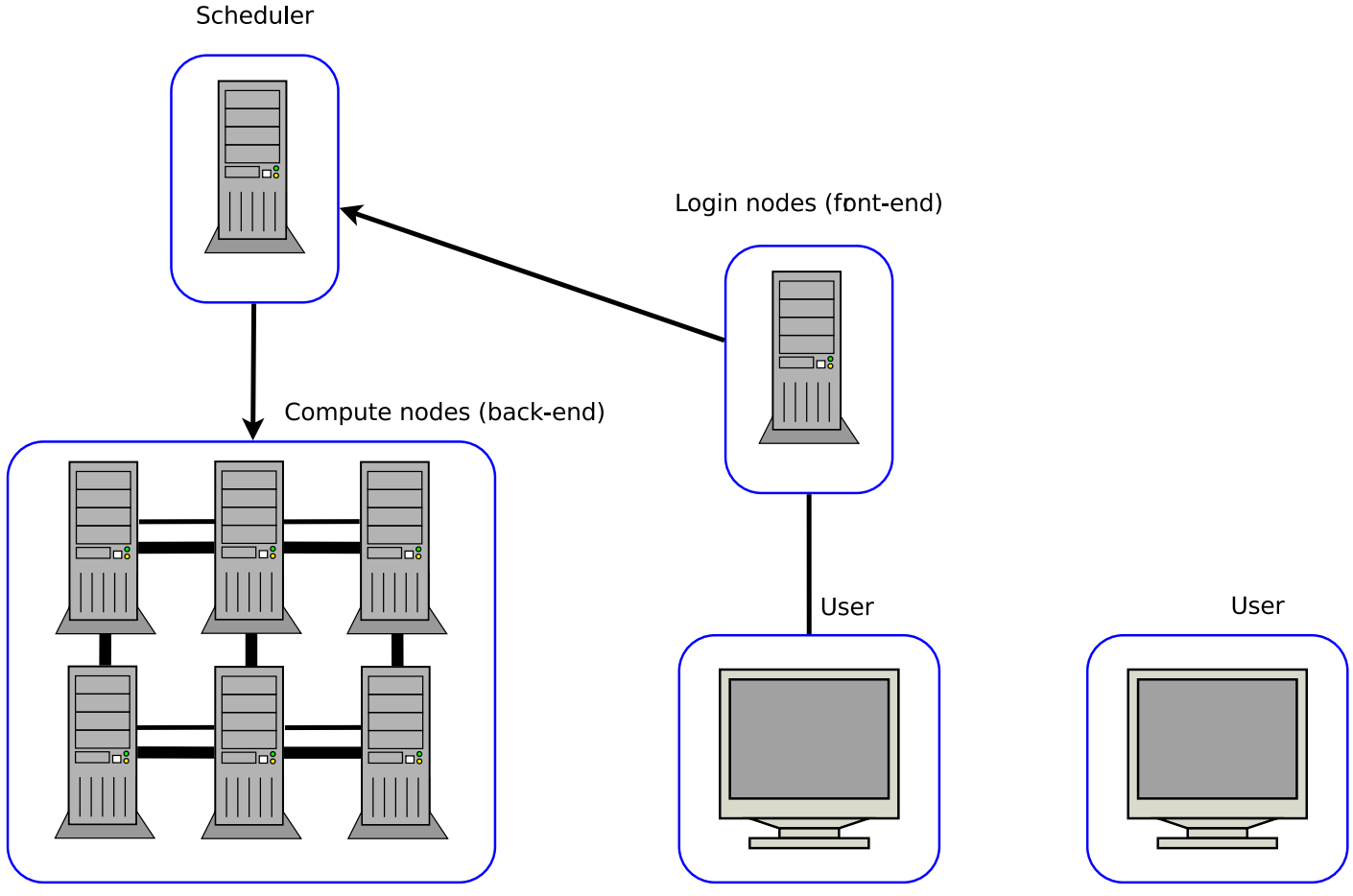
What is a cluster?



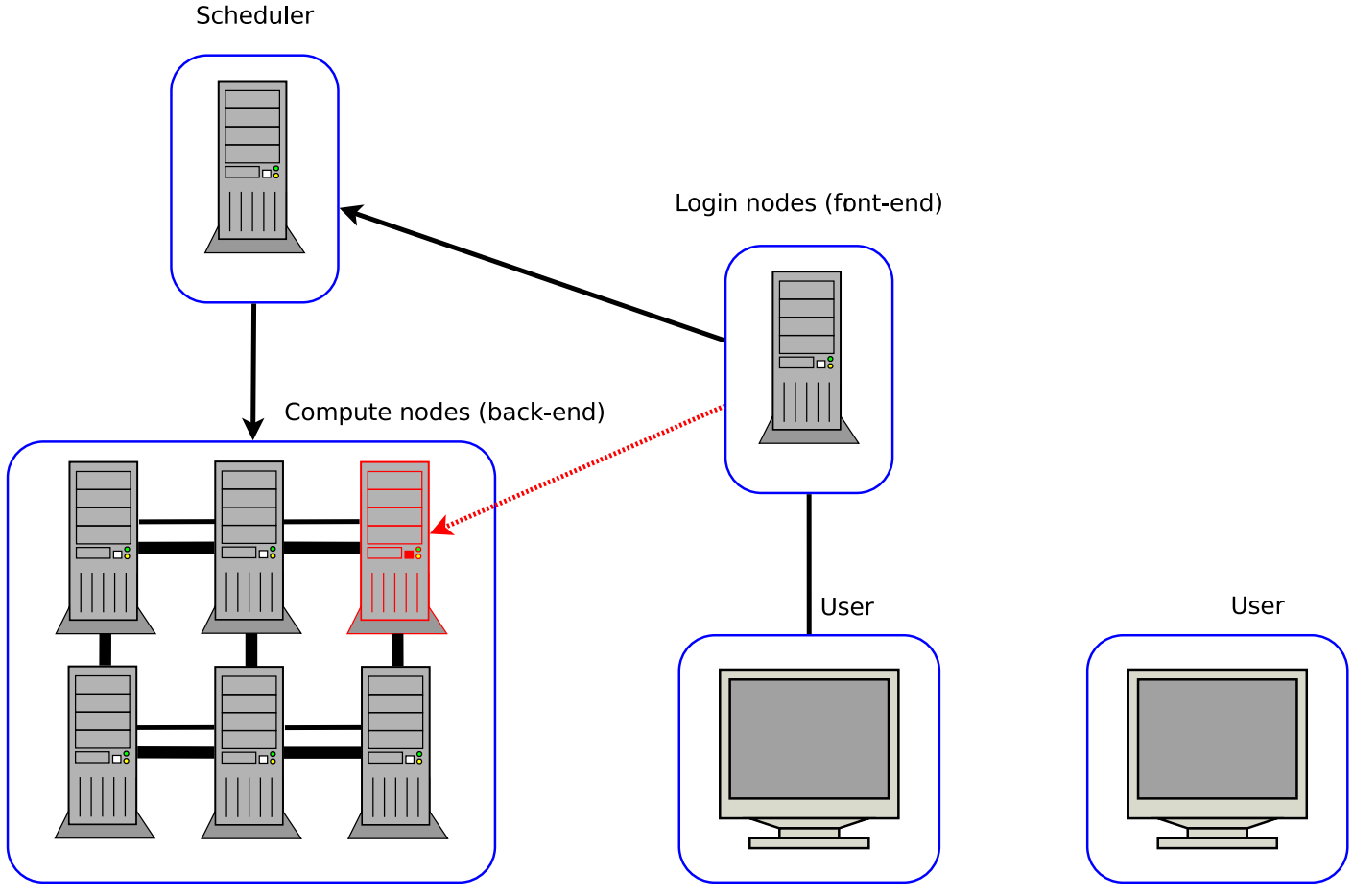
What is a cluster?



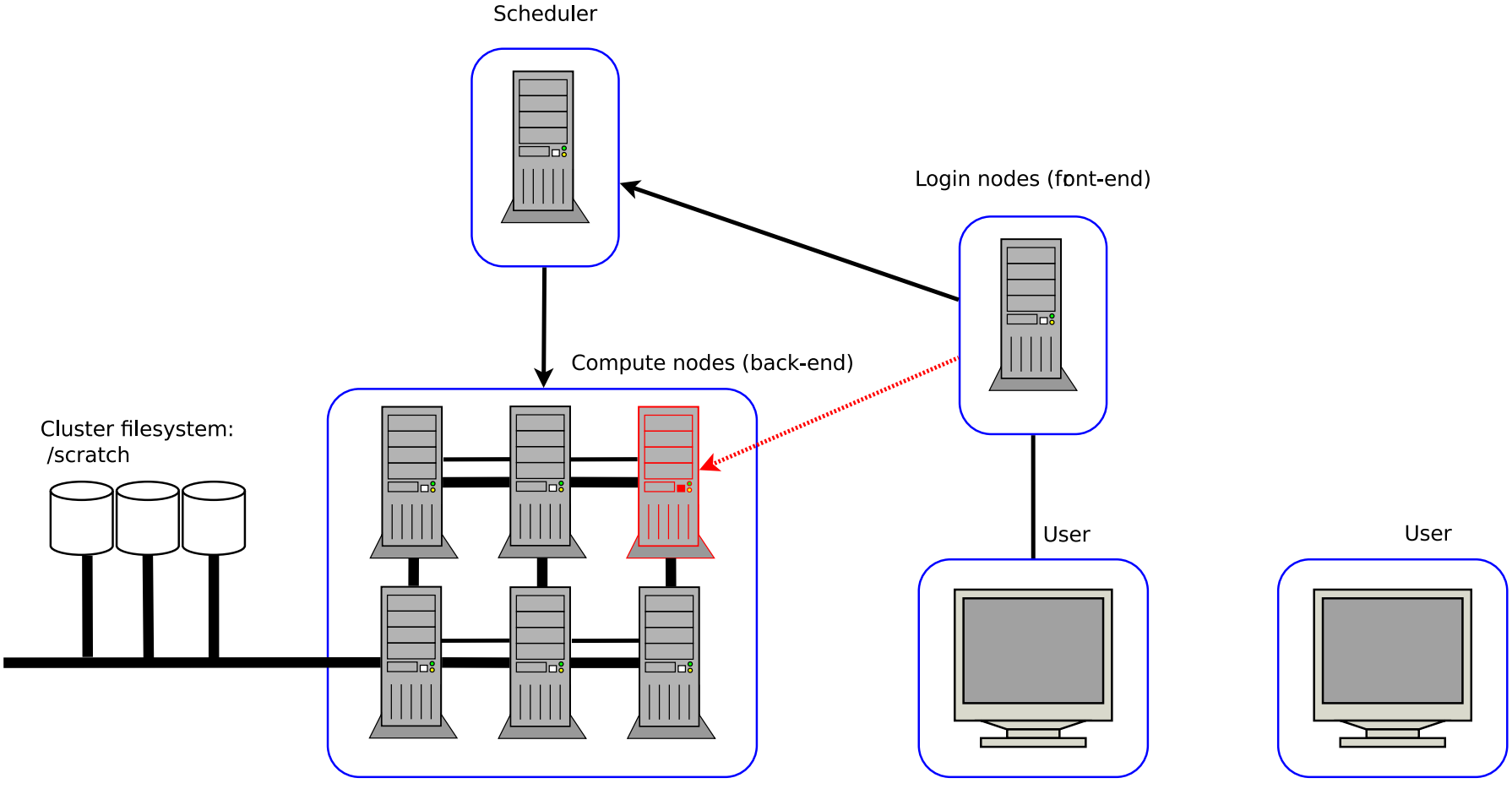
What is a cluster?



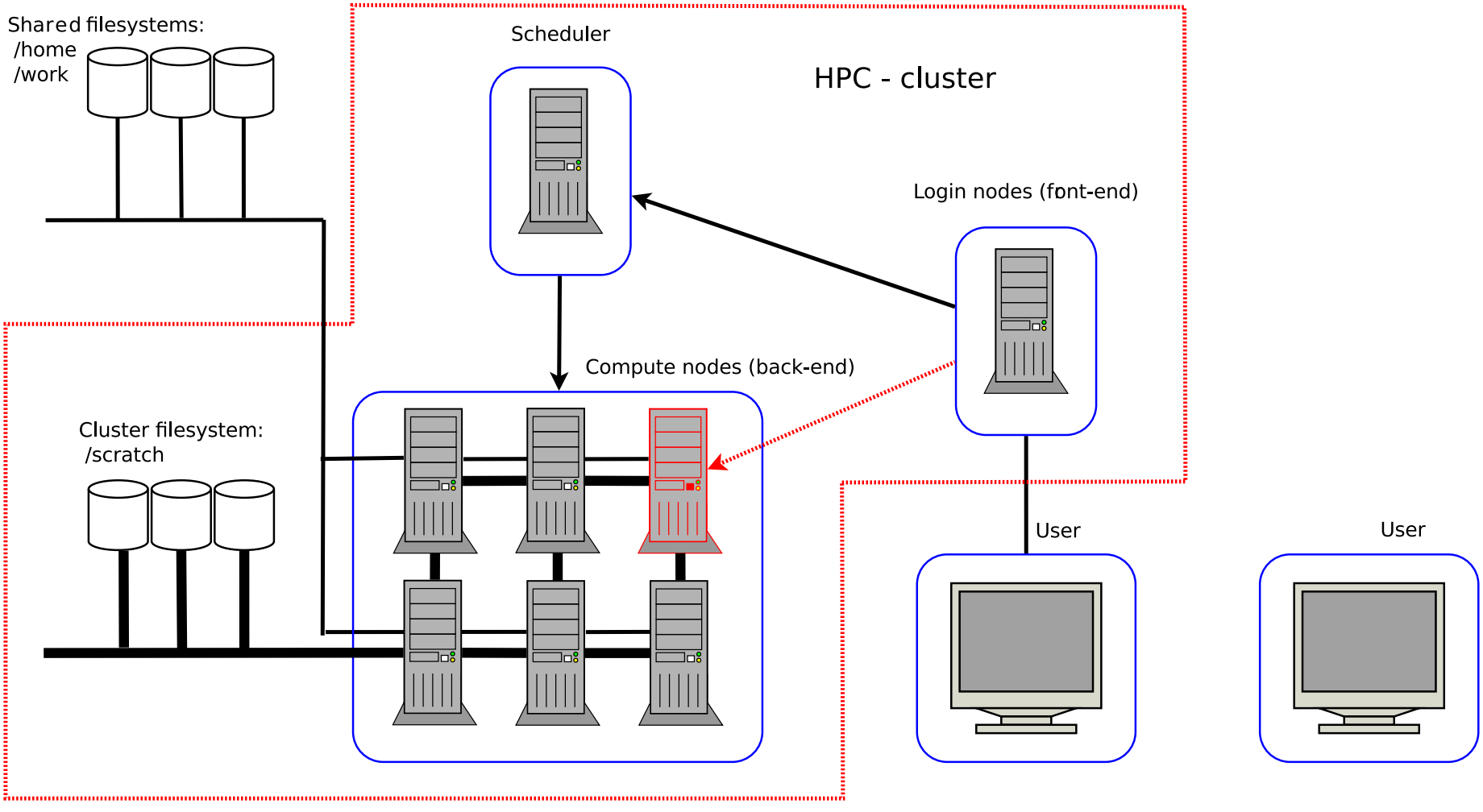
What is a cluster?



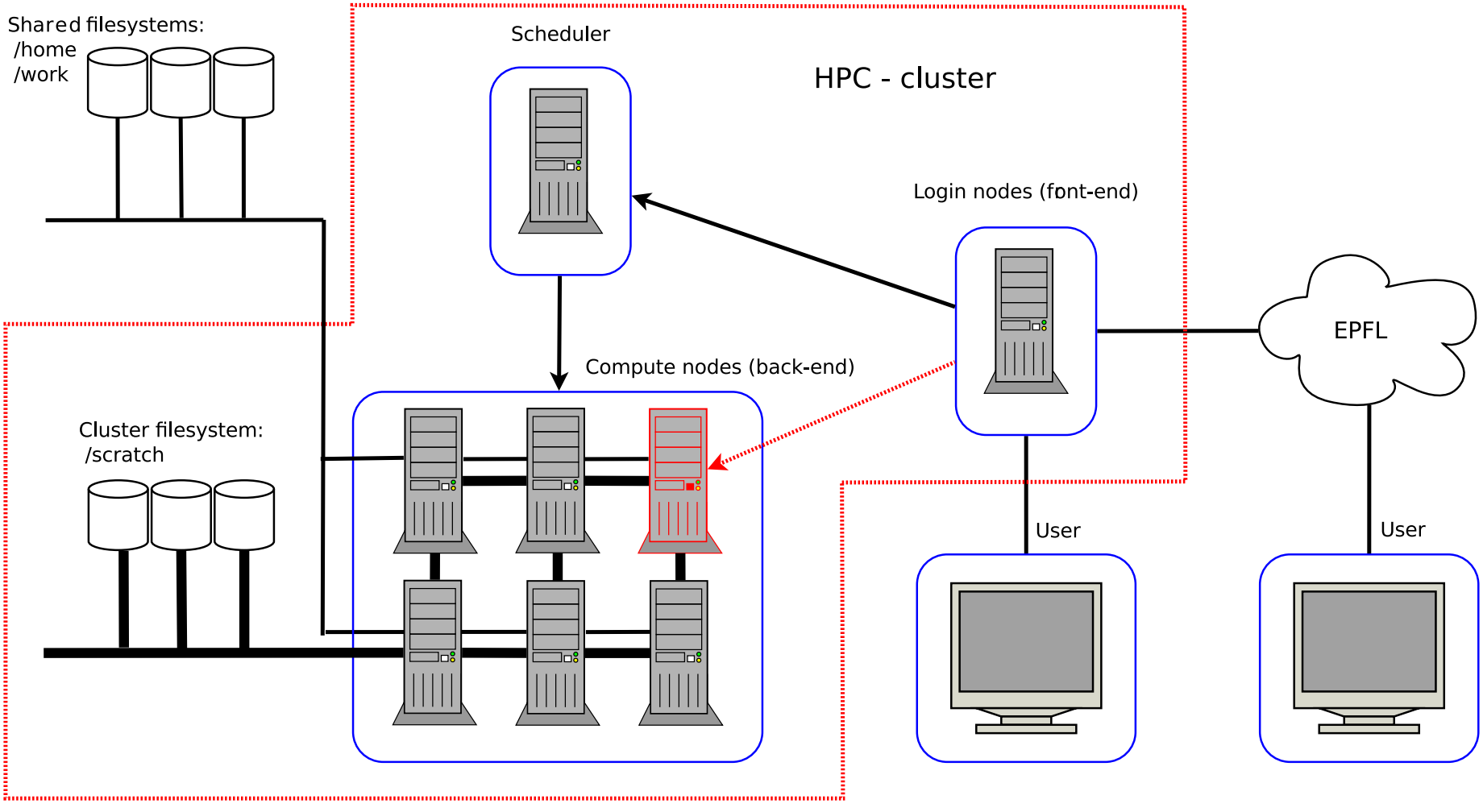
What is a cluster?



What is a cluster?



What is a cluster?



Our clusters

Login hostname	**Hosts** #	**Cores** # x GHz	**RAM** GB	**Network** Gbit/s	**Storage** TB	**Arch**
`deneb{1,2}.epfl.ch`	376	16x2.6	64	40 (IB)	350	E5v2
	144	24x2.5				E5v3
	16	16x2.6	+ *4x NVidia K40*			E5v2
	8	16x2.6	256			E5v2
	2	32x2.6	512			E5v2
`fidis.epfl.ch`	335	28x2.6	128	56 (IB)	375	E5v4
	72		256			E5v4
	216	28x2.6	192	100 (IB)		s6g1

What is each cluster optimized for?

Distinctive characteristics

- Network: Fast *vs.* Slow, Low latency *vs.* High latency
- Storage: Standard *vs.* High performance parallel
- Accelerators: GPUs, Xeon Phi, etc...

Parallel multi-node workloads: `deneb` and `fidis`

- Network: fast low latency network interconnect (Infiniband)
- Storage: fast parallel storage system (GPFS)

Single node and serial workload: `deneb` *`serial`*

- Network: 10Gb/s Ethernet

Shared Storage (cluster)

/scratch

- high performance temporary space
- not backed up
- low redundancy, built for performance
- local to each cluster
- automatically cleanup procedure deletes files (older than 2 weeks) without warning (when occupancy reaches a threshold)
- **for disposable files: intermediary results, temporary files**

Shared Storage (global)

/home

- per user quotas of 100GB
- backed up to a remote site
- available on all clusters
- **for important files: source code, final results, theses**

/work

- per group quotas
- 50GB for free
- for more: 300CHF/TB for 3 years (+ 100CHF/TB for backup)
- available on all clusters
- **for common files: software, datasets**

Connecting to a cluster [Hands-on!]

`ssh username@deneb2.epfl.ch`

- Linux: simply connect using `ssh`
- Windows: install git-bash, connect using `ssh` from git-bash
- OSX: connect using `ssh`

Basic shell commands, moving around

- `id`
- `pwd`
- `ls /scratch/<username>`
- `cd /scratch/<username>`

Batch Systems

Batch

Goal: to take a list of jobs and execute them according to a priority when appropriate resources become available
Interactive use is possible but it not the principal way of running jobs!

SLURM

We use SLURM on all our clusters. It's widely used in the HPC world and open source.
`http://slurm.schedmd.com`

sbatch

sbatch

The fundamental command is `sbatch` which submits jobs to the batch system.

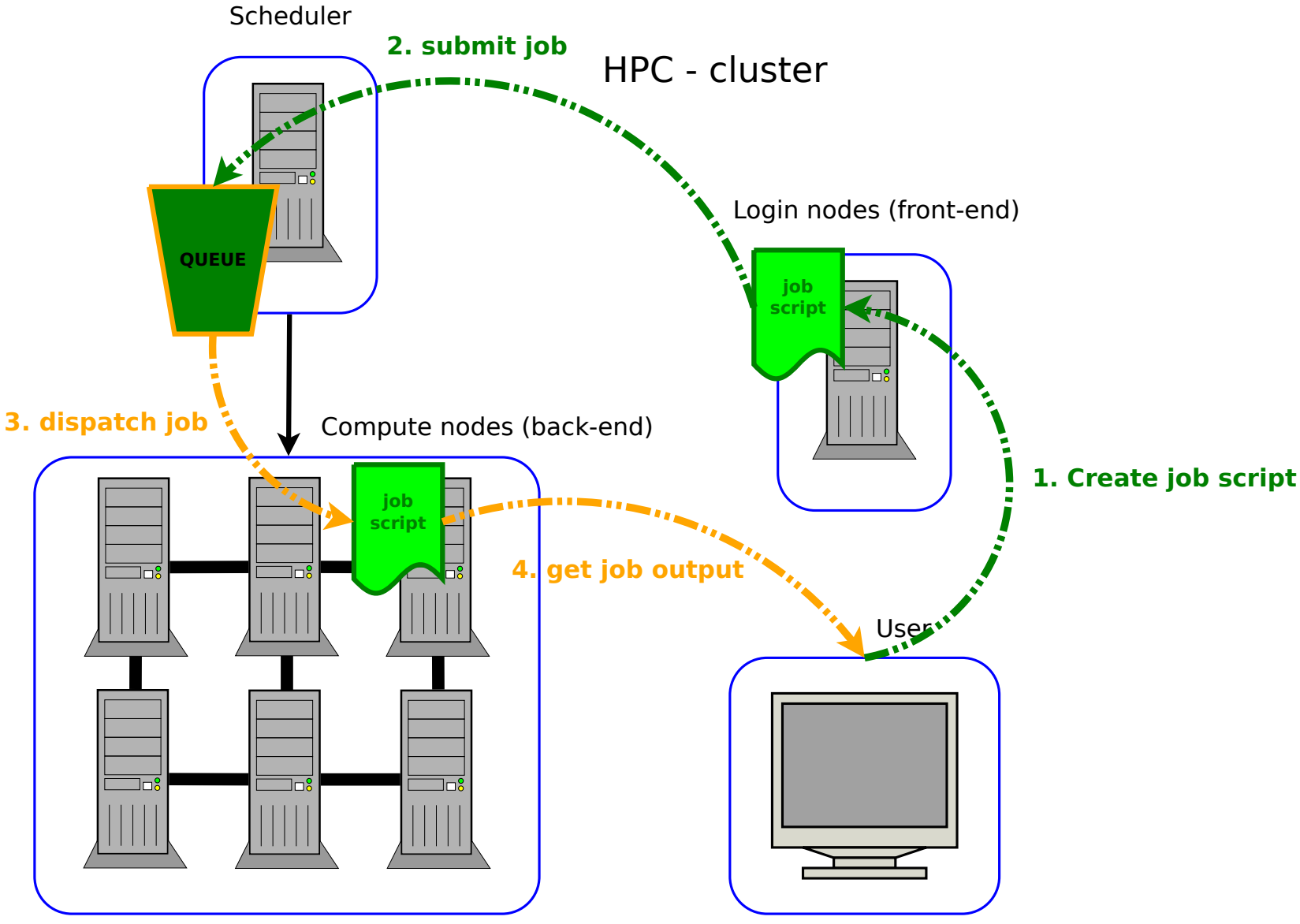
Worflow

A typical workflow to get your computation done is:

- create a short job-script
- submit it to the batch system
- *it will get executed*
- look at the output

The job **will wait in the queue** until resources are available to run it.

Workflow of a job



Exercise 1: sbatch [Hands-on!]

Prerequisites

Copy the examples to your working directory:

```
cp -r /scratch/examples/using-the-clusters .
```

Open and edit the first exercise

Open the file ex1.sh with your editor of choice:

- `nano`
- `emacs`
- `vim`
- `gedit`
- `...`

Exercise 1: sbatch [Hands-on!]

ex1.sh

```
#!/bin/bash
#SBATCH --workdir /scratch/<put-your-username-here>
#SBATCH --nodes 1
#SBATCH --ntasks 1
#SBATCH --cpus-per-task 1
#SBATCH --mem 1G
#SBATCH --partition serial
#SBATCH --account scitas-courses
#SBATCH --reservation intro2clusters

sleep 10
echo "hello from $(hostname)"
sleep 10
```

Exercise 1: SLURM directives

#SBATCH: directive to the batch system

```
--nodes 1
# the number of nodes to use - on Castor this is limited to 1

--ntasks 1
# the number of tasks (in an MPI sense) to run per job

--cpu-per-task 1
# the number of cores per aforementioned task

--mem 4096
# the memory required per node in MB

--time 12:00:00
--time 2-6
# the time required (12 hours and 2 days 6 hours respectively)

--partition build
# select a particular partition serial, parallel, debug, build
```

Exercise 1: submit ex1.sh to the batch system [Hands-on!]

No time was specified so it defaults to 15 minutes

```
$  sbatch ex1.sh
Submitted batch job 1509281

$ cat /scratch/<username>/slurm-1509281.out
hello from r02-node12
```

Remember the Job ID

The number returned by `sbatch` is known as the **Job ID** and is the unique identifier for a task. If you ever need to ask for help you'll need to know this number.

Exercise 1: What went on? [Hands-on!]

Sjob job-id

```
$ Sjob 1451763

       JobID     JobName     Cluster     Account  Partition  Timelimit       User      Group
------------ ------------ ----------- ----------- ---------- ---------- ---------- ----------
1451763             ex1.sh       deneb  scitas-ge      debug   00:15:00    rmsilva scitas-ge
1451763.bat+        batch        deneb  scitas-ge

               Submit             Eligible               Start                 End
------------------- ------------------- ------------------- -------------------
2017-09-12T11:39:52 2017-09-12T11:39:52 2017-09-12T11:39:52 2017-09-12T11:40:12
2017-09-12T11:39:52 2017-09-12T11:39:52 2017-09-12T11:39:52 2017-09-12T11:40:12

    Elapsed ExitCode      State
---------- -------- ----------
   00:00:20      0:0  COMPLETED
   00:00:20      0:0  COMPLETED

      NCPUS   NTasks        NodeList     UserCPU  SystemCPU     AveCPU  MaxVMSize
---------- -------- --------------- ---------- ---------- ---------- ----------
        16           r02-node02  00:00.002  00:00.025
        16        1  r02-node02  00:00.002  00:00.025   00:00:00    145216K
```

Cancelling jobs

scancel

To cancel a specific job:

```
scancel <JOB_ID>
```

To cancel all your jobs:

```
scancel -u <username>
```

To cancel all your jobs that are not yet running:

```
scancel -u <username> -t PENDING
```

Exercise 2: squeue [Hands-on!]

59GB of memory required

```
#!/bin/bash
#SBATCH --workdir /scratch/<username>
#SBATCH --nodes 1
#SBATCH --ntasks 1
#SBATCH --cpus-per-task 8
#SBATCH --mem 59G
#SBATCH --time 00:30:00
#SBATCH --partition serial
#SBATCH --account scitas-courses
#SBATCH --reservation intro2clusters


cd /scratch/examples/linpack/
./runme_xeon64
```

Exercise 2: what's going on? [Hands-on!]

squeue

With no arguments squeue will list all jobs currently in the queue! The output and information shown can be refined somewhat by giving options.

- `squeue -t R -u username`
- `squeue -t PD -u username`
- `squeue -t PD -u username --start`

Squeue

Squeue is an custom squeue that shows only your jobs with more useful information.

scontrol: more details

`scontrol -dd show job <jobid>`

Software

Why is a module system needed

- The OS version is restricted to an older one due to compatibility requirements of storage systems and specialized interconnects.
- The above is often in direct conflict with the needs of the HPC community, for which newer versions bring performance improvements and support for newer hardware (new CPU features).
- Many scientific codes are not even packaged under most Linux distributions.

Lmod

- **Lmod** is a utility that allows multiple, often incompatible, tools and libraries to co-exist on a system.

Modules [Hands-on!]

How software modules are organised

- Packages are organized hierarchically: `Compiler / MPI / blas`
- **Lmod** is designed to maintain the environment consistent
- **Lmod** does everything possible to automatically reload any software when one of the hierarchy layers is changed

Basic commands

- `module av(ailable)`
- `module load / unload <module-name>`
- `module spider <name>`
- `module save / restore <mnemonic-name>`
- `module purge`

Exercise 3: modules [Hands-on!]

ex3.sh: using module files

```
#!/bin/bash
#SBATCH --ntasks 1
#SBATCH --cpus-per-task 1
#SBATCH --nodes 1
#SBATCH --mem 4G
#SBATCH --time 00:05:00
#SBATCH --partition serial
#SBATCH --account scitas-courses
#SBATCH --reservation intro2clusters

echo STARTING AT $(date)

module purge
module load matlab

matlab -nodesktop -nojvm -r mymfile

echo FINISHED AT $(date)
```

Exercise 4: MPI parallel jobs [Hands-on!]

ex4.sh: srun to run parallel jobs

```
#!/bin/bash

#SBATCH --nodes 2
#SBATCH --ntasks 32
#SBATCH --cpus-per-task 1

module purge
module load intel
module load intel-mpi

cd osu
make clean
make

srun ./osu_allgather
```

Interactive access

Why interactive?

For debugging or running applications such as Matlab interactively we don't want to submit a batch job.

Sinteract or salloc

There are two main ways of getting access depending on what you want to achieve:

- `salloc` - standard tool for an interactive allocation
- `Sinteract` - custom tool to access a node

Behind the scenes both use the same mechanism as `sbatch` to get access to resources.

Sinteract

Sinteract --help

```
usage: Sinteract [-c cores] [-n tasks] [-t time] [-m memory]
[-p partition] [-a account] [-q qos] [-g resource] [-r reservation]
options:
  -c cores        cores per task (default: 1)
  -n tasks        number of tasks (default: 1)
  -t time         as hh:mm:ss (default: 00:30:00)
  -m memory       as #[K|M|G] (default: 4G)
  -p partition    (default: parallel)
  -a account      (default: scitas-ge)
  -q qos          as [normal|gpu|gpu_free|mic|...] (default: )
  -g resource     as [gpu|mic][:count] (default is empty)
  -r reservation reservation name (default is empty)
  -s contraints  list of required features (default is empty)
                     Deneb/Eltanin: E5v2, E5v3
                     Fidis/Gacrux: E5v4, s6g1, mem128gb, mem192gb, mem256g
examples:
    /usr/bin/Sinteract -c 16 -p serial
    /usr/bin/Sinteract -p gpu -q gpu_free -g gpu:1
```

Sinteract [Hands-on!]

Sinteract

```
[<user>@deneb2 ~]$ Sinteract -p debug
Cores:              1
Tasks:              1
Time:               00:30:00
Memory:             4G
Partition:          debug
Account:            scitas-ge
Jobname:            interact
Resource:
QOS:                scitas
Reservation:

salloc: Granted job allocation 1451785
salloc: Waiting for resource configuration
salloc: Nodes r02-node02 are ready for job
[<user>@r02-node02 ~]$
```

salloc [Hands-on!]

salloc then srun for MPI tasks

```
[<user>@deneb2 ]$ salloc -N 1 -n 2 --mem 2048 -p debug
salloc: Granted job allocation 1451788
salloc: Waiting for resource configuration
salloc: Nodes r02-node02 are ready for job

[<user>@deneb2 ]$ hostname
deneb2

[<user>@deneb2 ]$ srun hostname
r02-node02
r02-node02

[<user>@deneb2 ]$ exit
salloc: Relinquishing job allocation 1451788
```

Sinteract: storage [Hands-on!]

Storage locations

The different storage locations are accessible via environment variables.

- `$SCRATCH` - your scratch folder on the current cluster `/scratch` filesystem
- `$TMPDIR` - a **temporary** folder in a **local** filesystem (generally `/tmp`)
- `$WORK` - your groups' folder in the global `/work` filesystem

Try it: Sinteract -p debug

```
[rmsilva@r02-node02 ~]$ echo $SCRATCH
/scratch/rmsilva
[rmsilva@r02-node02 ~]$ echo $TMPDIR
/tmp/1451790
[rmsilva@r02-node02 ~]$ echo $WORK
/work/scitas-ge
```

The debug partition

`--partition=debug`

All the clusters have a few nodes that only allow short jobs and are intended to give you quick access to debug jobs:

- `salloc -p debug`
- `#SBATCH -p debug`
- `Sinteract -p debug`

The number of nodes in the partition and the limits vary by cluster.

Fairshare

Not everyone is equal

A group's priority on the clusters is related to to the number of shares (percentage of the cluster) that they have committed to use (paid) and their recent usage relative to their shares.

Within a group the relative consumption of the members decides who has more priority.

`http://slurm.schedmd.com/fair_tree.html`
`http://slurm.schedmd.com/priority_multifactor.html`

Fairshare

Current usage and priority (truncated)

```
        $ Sshare
             Account       User  RawShares  NormShares    RawUsage   NormUsage  EffectvUsage  FairShare
-------------------- ---------- ---------- ----------- ----------- ----------- ------------ ----------

             Account       User  RawShares  NormShares    RawUsage   NormUsage  EffectvUsage  FairShare
-------------------- ---------- ---------- ----------- ----------- ----------- ------------ ----------
scitas-ge                               1    0.005988       89953    0.000198     0.000515
 scitas-ge                aubort         1    0.062500           0    0.000000     0.000000   0.195841
 scitas-ge              clemenco         1    0.062500       12486    0.000027     0.138814   0.171577
 scitas-ge                 culpo         1    0.062500         186    0.000000     0.002074   0.173310
 scitas-ge              ddossant         1    0.062500           0    0.000000     0.000000   0.195841
 scitas-ge              degiorgi         1    0.062500           0    0.000000     0.000000   0.195841
 scitas-ge              desbioll         1    0.062500           0    0.000000     0.000000   0.183709
 scitas-ge                eroche         1    0.062500           0    0.000000     0.000000   0.195841
 scitas-ge              foureste         1    0.062500           0    0.000000     0.000000   0.181976
 scitas-ge              guglielm         1    0.062500           0    0.000000     0.000000   0.195841
 scitas-ge               nvarini         1    0.062500           0    0.000000     0.000000   0.195841
 scitas-ge                 query         1    0.062500           1    0.000000     0.000012   0.178510
 scitas-ge              rezzonic         1    0.062500          80    0.000000     0.000897   0.175043
 scitas-ge               richart         1    0.062500           1    0.000000     0.000019   0.176776
 scitas-ge               rmsilva         1    0.062500           0    0.000000     0.000004   0.180243
 scitas-ge             scitasbui+        1    0.062500       77195    0.000170     0.858180   0.169844
 scitas-ge                vkeller        1    0.062500           0    0.000000     0.000000   0.195841
```

Level FS shows you if you are under (>1) or over (<1) served.

Helping yourself

`man` pages are your friends!

- `man sbatch`
- `man sacct`
- `man gcc`
- `module load intel; man ifort`

Getting help

1234@epfl.ch

- send a mail to 1234@epfl.ch
- start the subect with **HPC**

We need to know as many of the following as possible

- the Job ID
- the directory location and name of the submission script
- where the "slurm-*.out" file is to be found
- how the "`sbatch`" command was used to submit it
- the output from "`env`" and "`module list`" commands

Going further

SCITAS offers courses in

- Compiling code and using MPI
- MPI, an introduction to parallel programming
- MPI, advanced parallel programming
- Introduction to profiling and software optimisation
- Computing on GPUs
- Data Management: code and large files
- Introduction to Linux

Useful links

links

Change your shell at:

`https://cadiwww.epfl.ch/cgi-bin/accountprefs/`

SCITAS web site:

`http://scitas.epfl.ch`

(in particular) SCITAS documentation space:

`http://scitas-data.epfl.ch/kb`

SLURM man pages:

`http://slurm.schedmd.com/man_index.html`