Consortium des Équipements de Calcul Intensif

How to use efficiently the storage solutions provided on the CECI clusters

Ariel Lozano

CÉCI HPC Training 2020

Prereqs

- To follow properly this presentation you must be already familiar with:

November 2020

- 12 Nov Damien François, "Preparing, submitting and managing jobs with Slurm"

October 2020

- 21 Oct Bernard Van Renterghem, "Introduction to modules and software on a CECI cluster"
- 20 Oct Damien François, "Introduction to scientific software development and deployment"
- 20 Oct Olivier Mattelaer, "Connecting with SSH from Windows: Introduction and advanced topics"
- 20 Oct Juan Cabrera, Olivier Mattelaer, "Connecting with SSH from Linux or Mac: Introduction and advanced topics"
- 19 Oct Bernard Van Renterghem, "Introduction to Linux and the command line"
- 19 Oct Frédéric Wautelet, "Introduction to high-performance computing"

Some context

- Nowadays the **best performant** ‘units’ of long term storage, PCIe SSDs, give ~2 GB/s of sequential read/write. This will go down to about ~400MB/s for random read/write of many small files.

- A basic sequential write test on my 2019 laptop with a consumer midrange SSD, SK Hynix PC601 NVMe 512GB

```
$ dd if=/dev/zero of=test2GBdump bs=1M count=2048; sync

2048+0 records in
2048+0 records out
2147483648 bytes (2.1 GB, 2.0 GiB) copied, 1.81005 s, 1.2 GB/s
```

- This is just a basic test with just a single task using the storage intensively. The CPU access the SSD via PCI express lanes.

Previous: HPC cluster

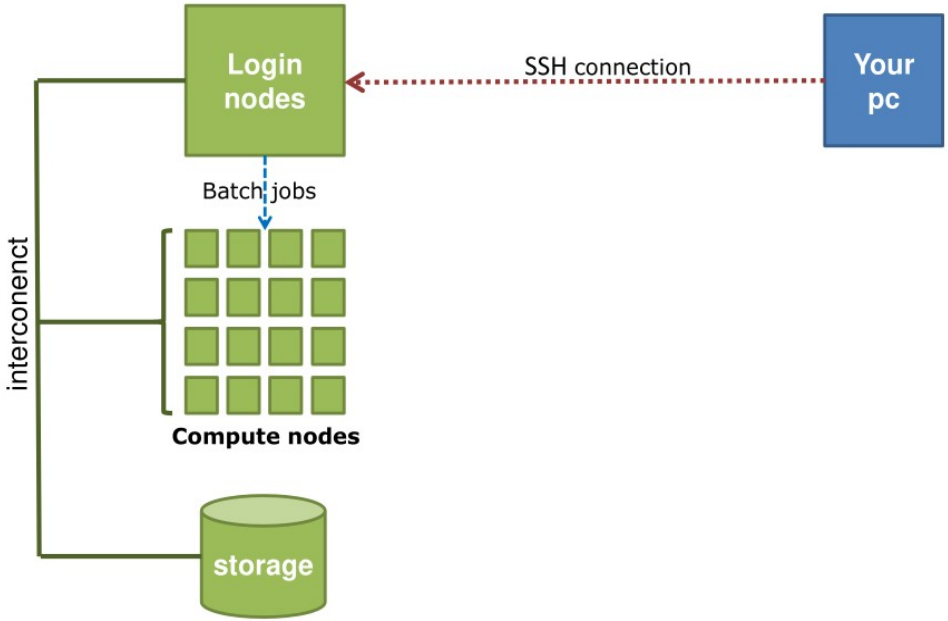
- A computer ‘cluster’ is a group of interconnected computers working together closely, so that in many respects they form a single computer

  Frédéric Wautelet, "Introduction to high-performance computing"

- Corollary: Access to **most** of the different storage solutions available on these systems happens via the network

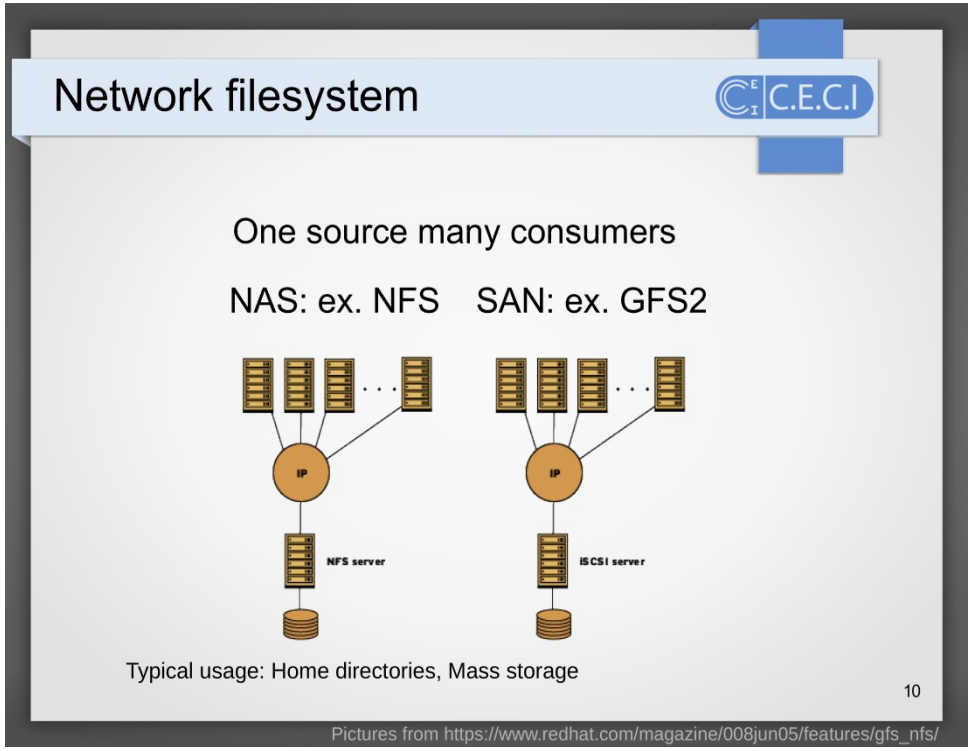
Previous: HPC cluster

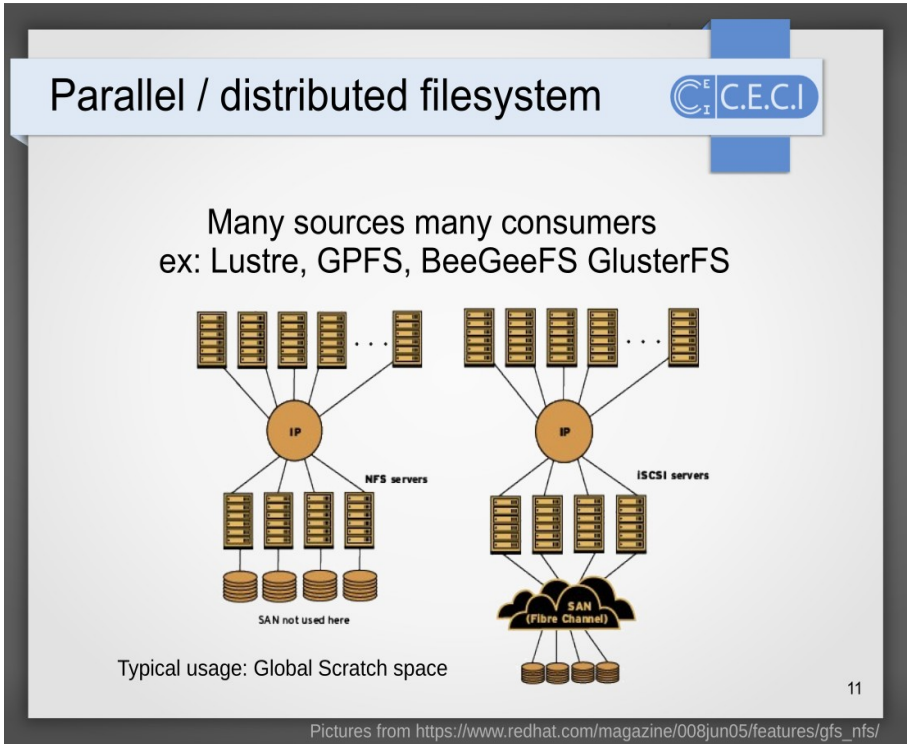
A cluster in a nutshell



Frédéric Wautelet, "Introduction to high-performance computing"

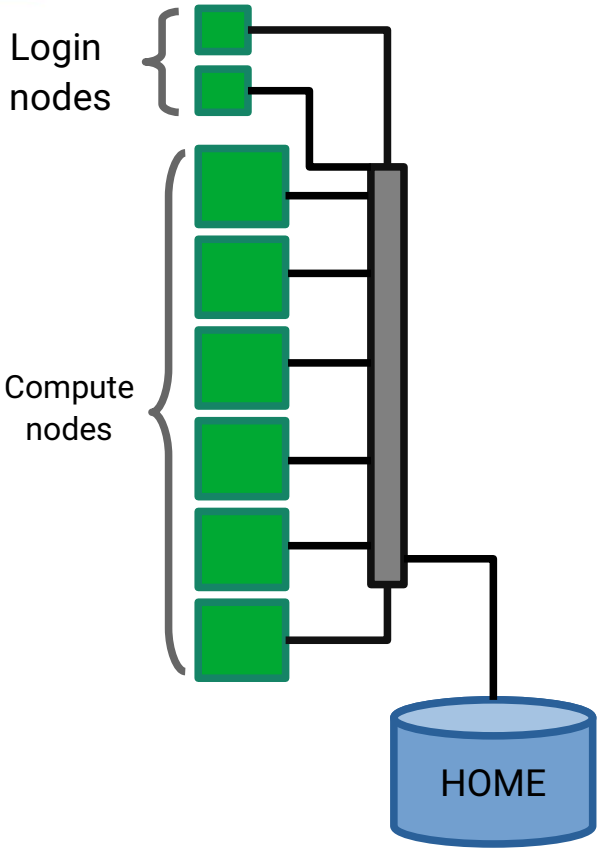
Previous: Network storage solutions





Damien François, "Introduction to data storage and access"

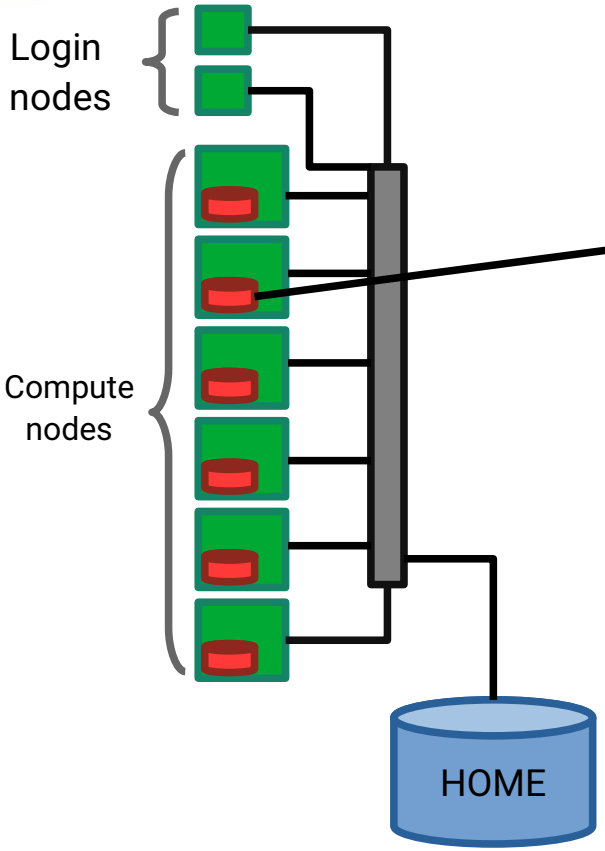
Storages on CECI clusters



$HOME

- Area where you land on login
- Network storage visible from all compute nodes and login
- Accessible via a **single** network link

Storages on CECI clusters



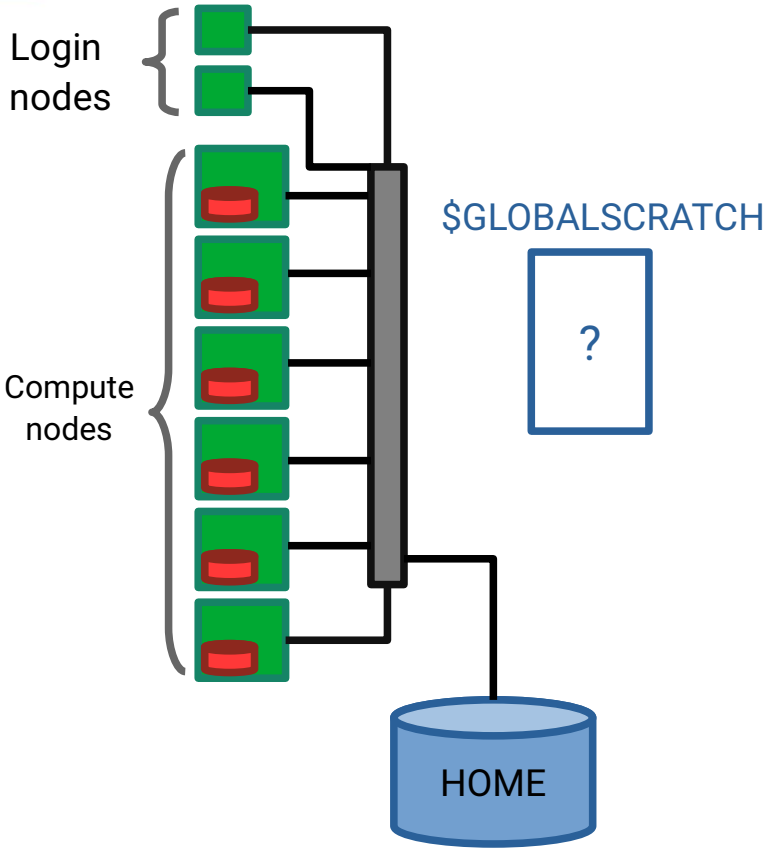
$HOME

- Area where you land on login
- Network storage visible from all compute nodes and login
- Accessible via a **single** network link

$LOCALSCRATCH

- Internal storage inside each compute node (HDD or SSD)
- Can provide high performance via a RAID0 setup
- Accessible **only** by the compute node hosting it
- Shared with other jobs running on the same node
- Usage only temporary during the job runtime
- Serial jobs or Parallel SMP (openmp, python multiprocessing)

Storages on CECI clusters



$HOME

- Area where you land on login
- Network storage visible from all compute nodes and login
- Accessible via a **single** network link

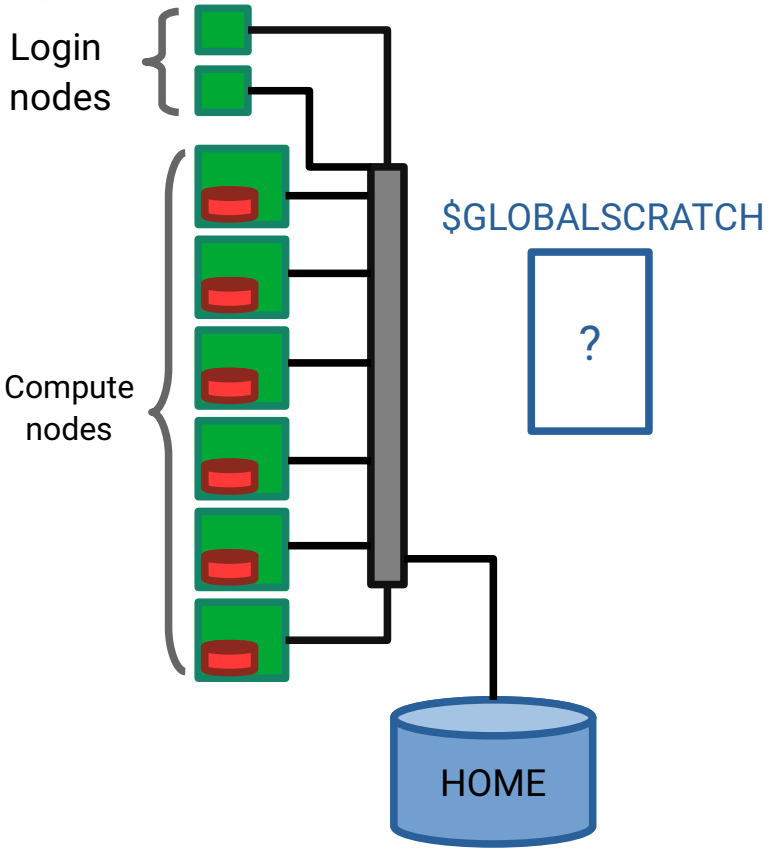
$LOCALSCRATCH

- Internal storage inside each compute node (HDD or SSD)
- Can provide high performance via a RAID0 setup
- Accessible **only** by the compute node hosting it
- Shared with other jobs running on the same node
- Usage only temporary during the job runtime
- Serial jobs or Parallel SMP (openmp, python multiprocessing)

$GLOBALSCRATCH

- Implemented via different setups
- Accessible by all compute nodes and login
- Accessible via a network interconnet
- Can be composed of a single or multiple storage sources
- Data there stays persistently (but all is removed in yearly maintenances)
- You must clenaup from time to time
- All jobs but **only option** for multinode-parallel jobs (big MPI jobs)

Storages on CECI clusters



How do we access these storage areas ?

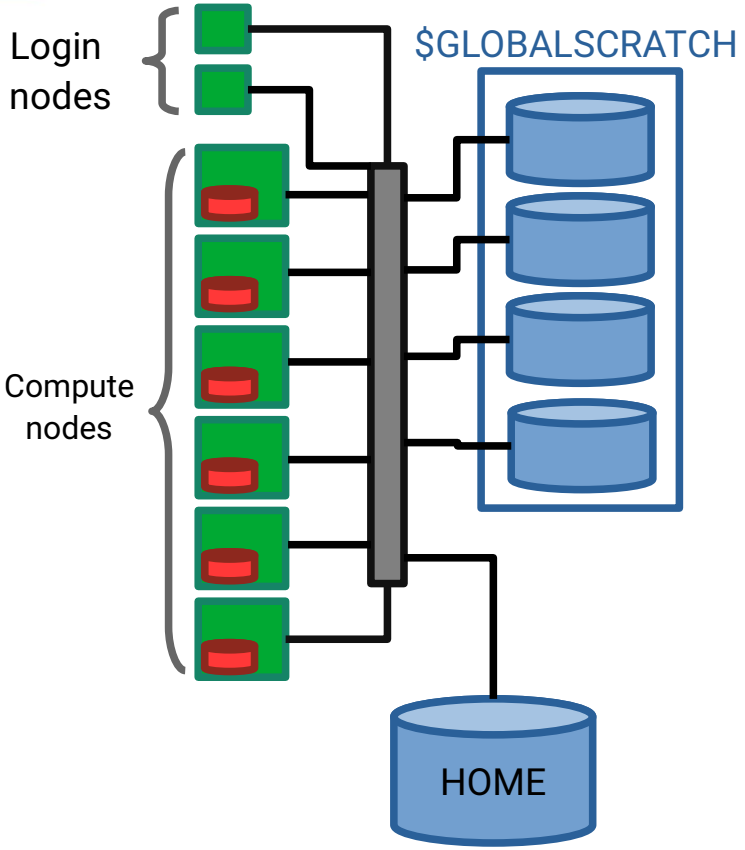
There are environment variables defined on the clusters pointing to them

- $HOME
- $LOCALSCRATCH
- $GLOBALSCRATCH

For LOCALSCRATCH as it's internal to each node, it can be accessed only by jobs submitted to a given node

Lemaitre3 (soon also NIC5)

Dedicated global parallel filesystem



$HOME
- 100GB quota

$LOCALSCRATCH
- Single SSD
- 200GB maximum capacity
- Data deleted when job finished!

$GLOBALSCRATCH
- Parallel filesystem distributed among multiple storage servers (BeeGFS)
- Accessible via multiples high speed network interconnet (100Gb/s)
- Visible as one single volume from login/compute nodes
- 570 TB size in total
- No quotas enforced (remember to cleanup)
- The storage will be fully purged on yearly maintenances

Lemaitre3 (soon also NIC5)

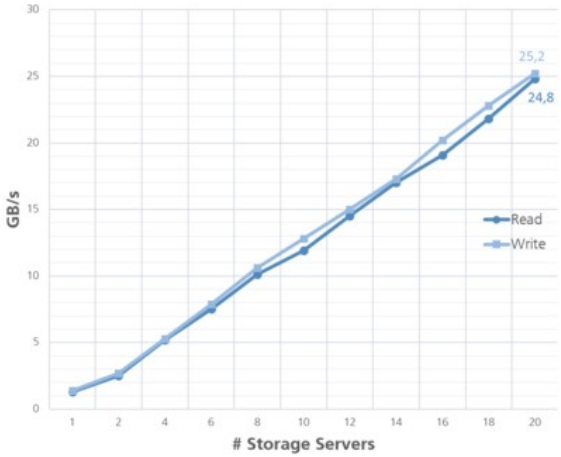
Dedicated global parallel filesystem

https://en.wikipedia.org/wiki/BeeGFS

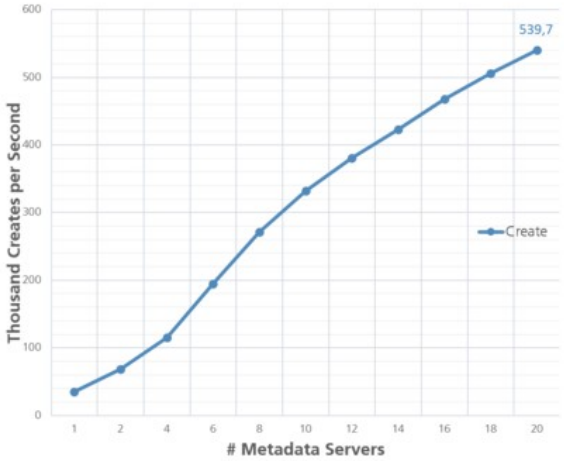
Benchmarks [edit]

The following benchmarks have been performed on Fraunhofer Seislab, a test and experimental cluster at Fraunhofer ITWM with 25 nodes (20 compute + 5 storage) and a three-tier memory: 1 TB RAM, 20 TB SSD, 120 TB HDD. Single node performance on the local file system without BeeGFS is 1,332 MB/s (write) and 1,317 MB/s (read).

The nodes are equipped with 2x Intel Xeon X5660, 48 GB RAM, 4x Intel 510 Series SSD (RAID 0), Ext4, QDR Infiniband and run Scientific Linux 6.3, Kernel 2.6.32-279 and FhGFS 2012.10-beta1.



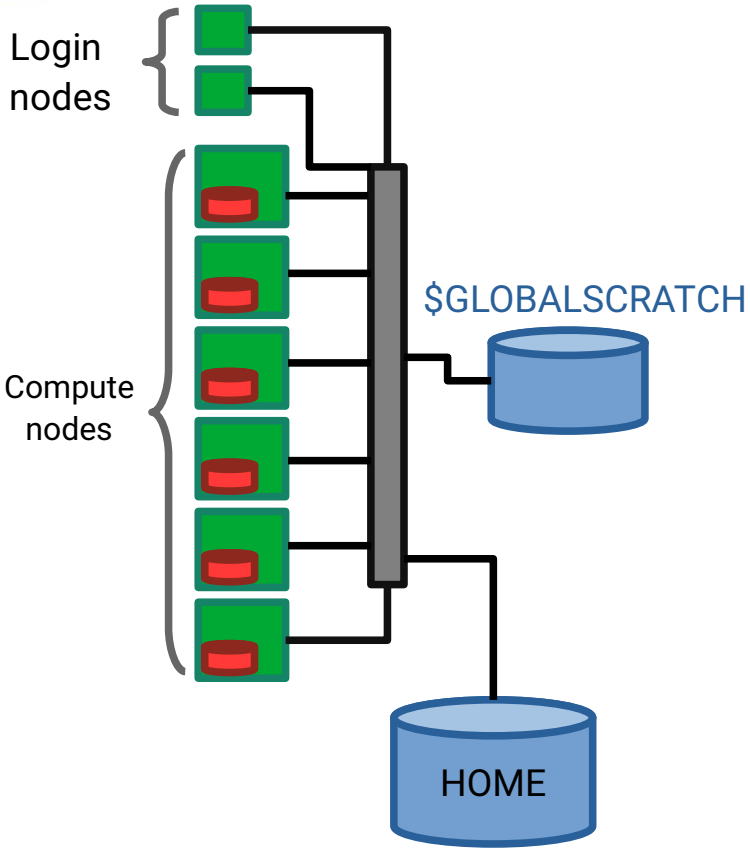
Read/Write Throughput



File Creates

https://www.beegfs.io/c/resources/
https://indico.mathrice.fr/event/5/session/5/contribution/12/material/slides/0.pdf

Hercules



$HOME

- 200GB quota

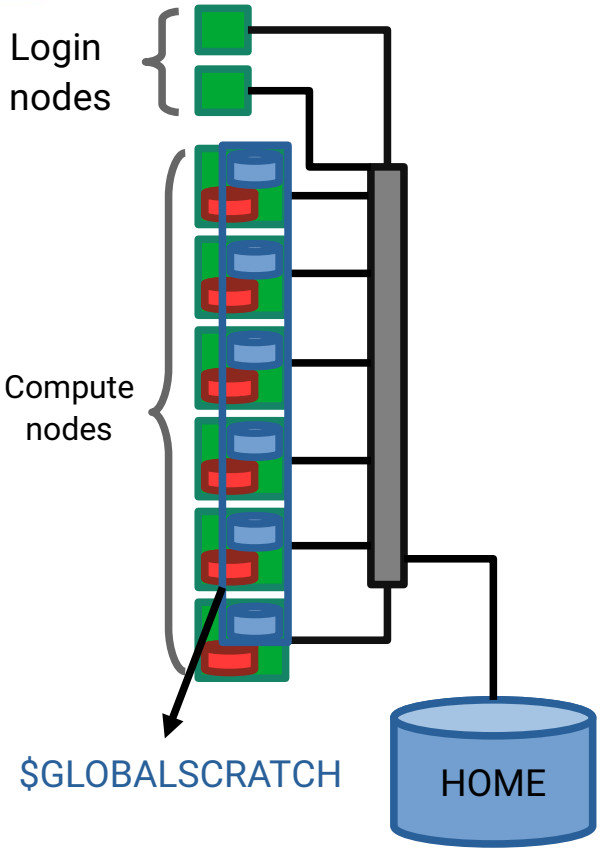
$LOCALSCRATCH

- RAID0 of 4 HDDs
- her2-w065...096: 1TB (features=intel)
- her2-w099...126: 4TB (features=amd)
- her2-w127...128: 8TB (only nodes with 2TB RAM)
- Data deleted when job finished!

$GLOBALSCRATCH

- Single storage server mounted by a NFS share
- Accessible via a single network link (10Gb/s)
- 400GB soft 4TB hard quota

Dragon2



$HOME

- 40GB quota

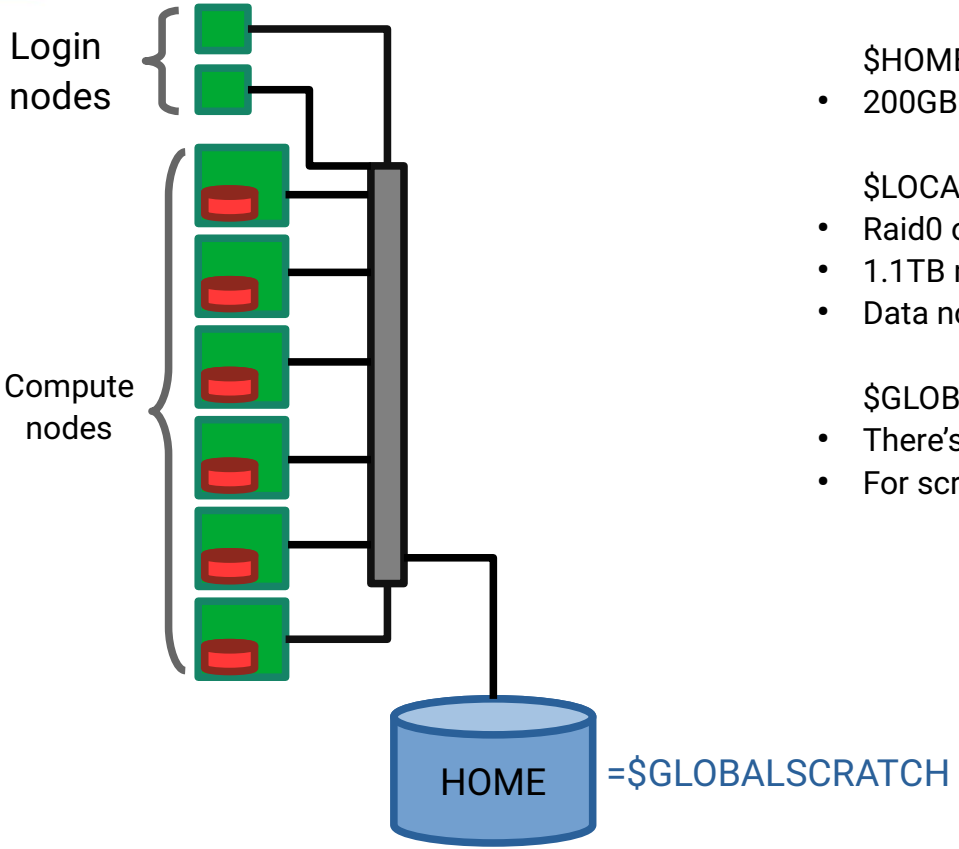
$LOCALSCRATCH

- Raid0 of 3 HDDs
- 3TB maximum capacity
- Data not deleted when job finished, please cleanup at end !

$GLOBALSCRATCH

- Parallel filesystem distributed among multiple storage targets (BeeGFS)
- A partition on each compute node is part to build the scratch
- Visible as one single volume from login/compute nodes
- 52 TB size in total
- Accessible via the same network interconnet as the nodes (10Gb/s)
- No hard quotas enforced (remember to cleanup)

Dragon1



$HOME

- 200GB quota

$LOCALSCRATCH

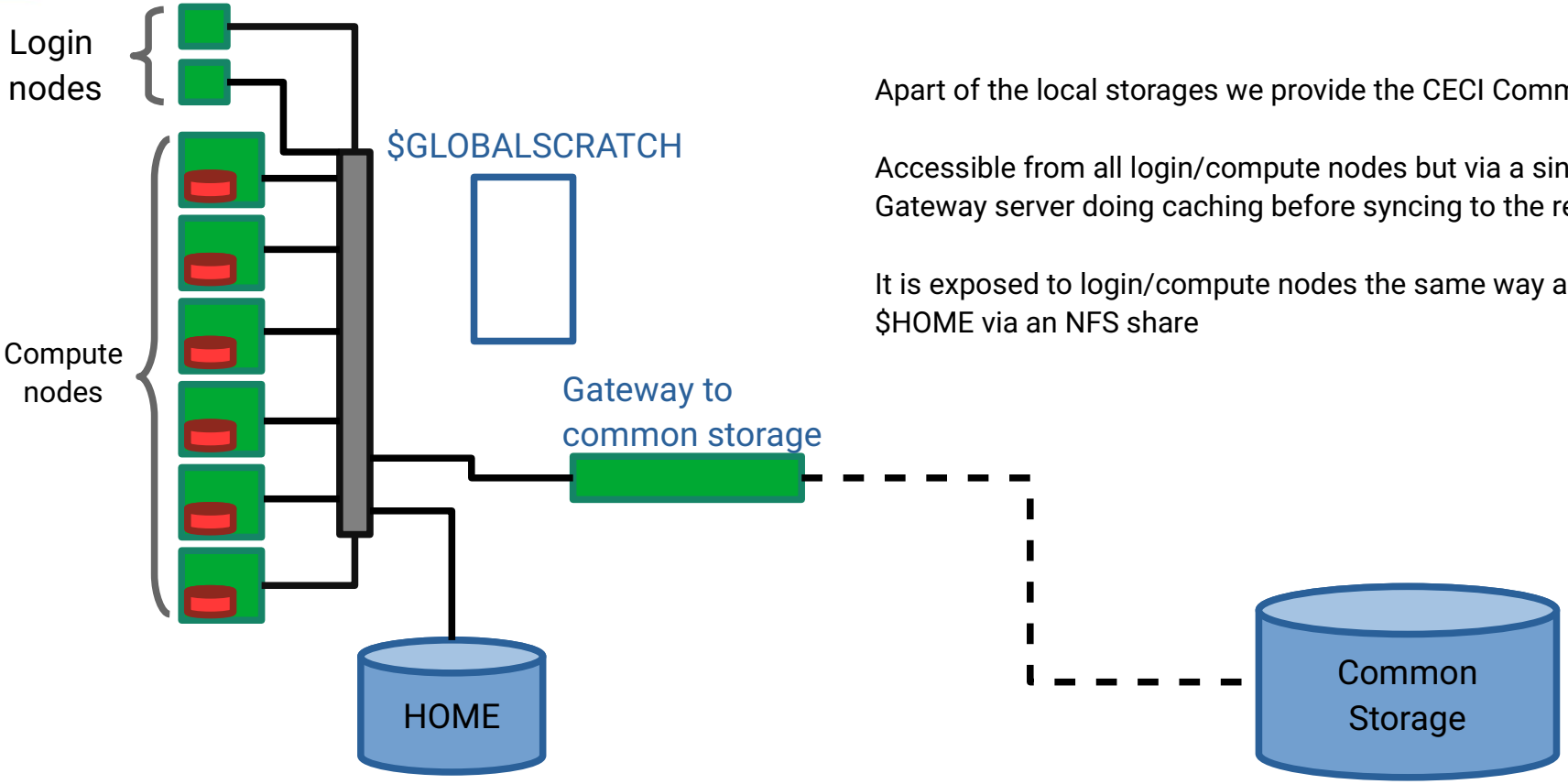
- Raid0 of 3 HDDs
- 1.1TB maximum capacity
- Data not deleted when job finished, please cleanup at end !

$GLOBALSCRATCH

- There's **no** dedicated $GLOBALSCRATCH
- For scrtips portability among CECI the variable points to HOME

CECI Common storage

external remote storage accesible by all clusters



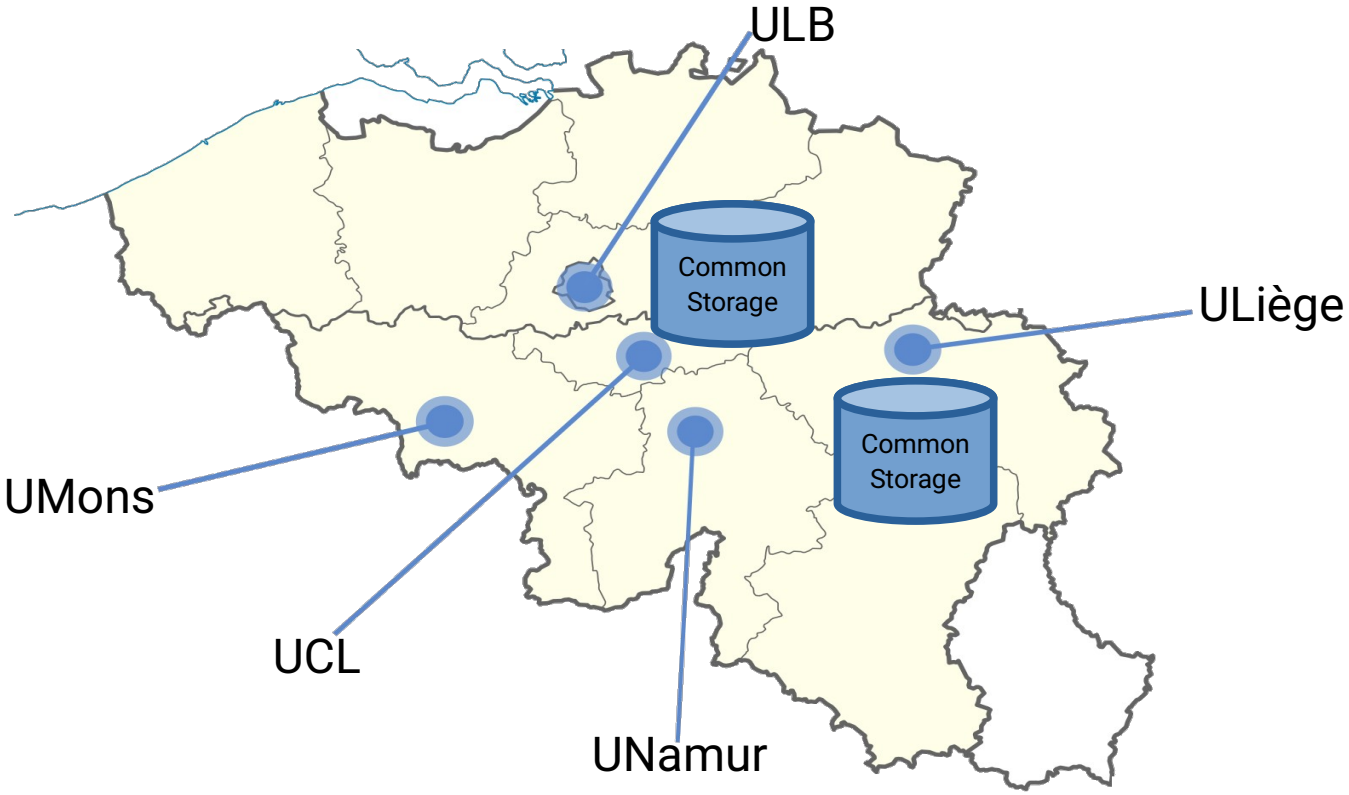
Apart of the local storages we provide the CECI Common Storage

Accessible from all login/compute nodes but via a single network link
Gateway server doing caching before syncing to the remote storage

It is exposed to login/compute nodes the same way as the $HOME via an NFS share

CECI Common storage

external remote storage accesible by all clusters



The main storage servers are in ULiège and UCL
There is a dedicated fiber among sites for this solution

CECI Common storage

external remote storage accesible by all clusters

/CECI/home
- Each user gets a personal area here by default
- Full personal path is pointed with $CECIHOME variable from any cluster
- Quota of 100GB

/CECI/proj
- Area where a team with a project can get a common folder for sharing data
- Must be requested by a PI
- Quota decided according to the project's needs

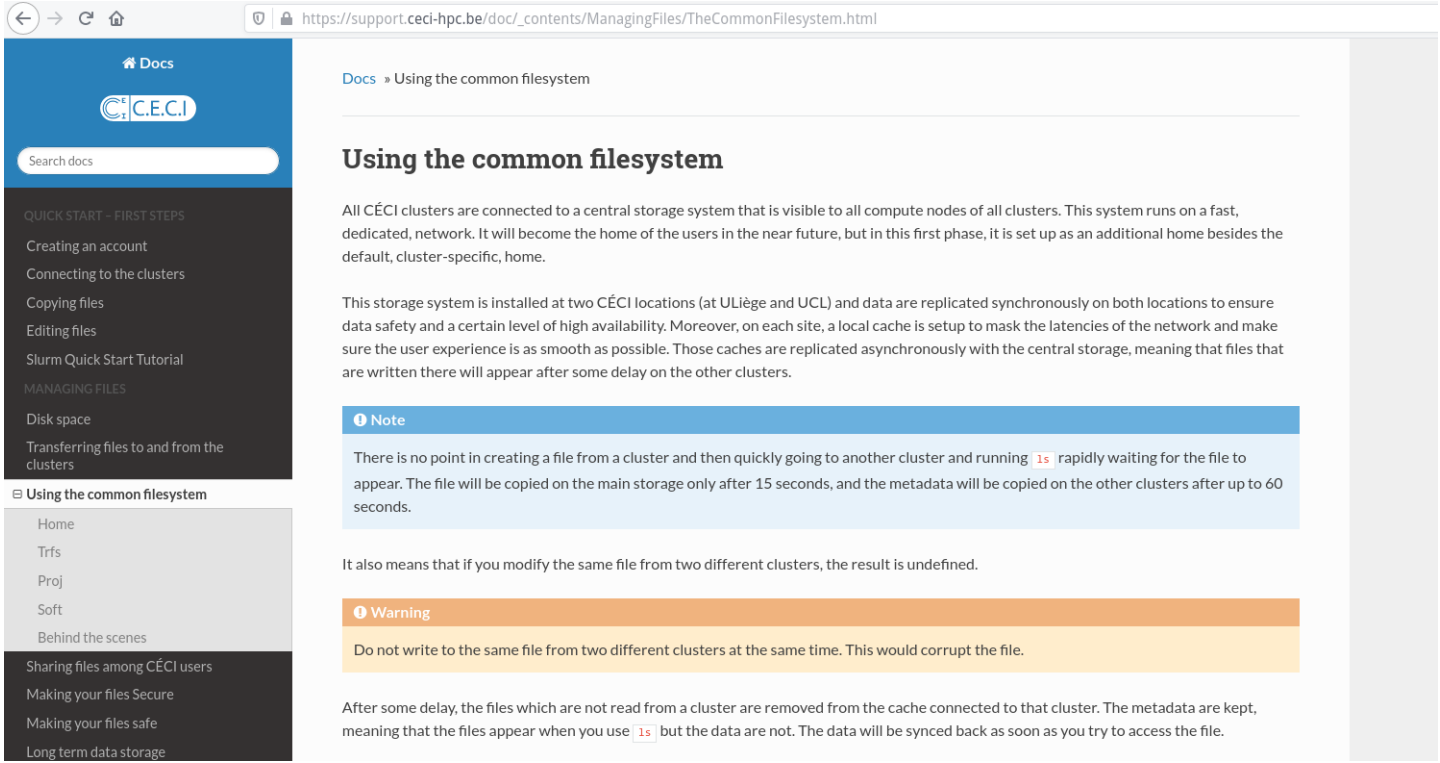
/CECI/trsf
- Area to be used to move big amounts of data between clusters
- Common area pointed with $CECITRSF (create your own subfolder)
- Meant only for **temporary** copying from one cluster to another
- Data here can be purged every 6 months
- Quota of 1TB soft 10TB hard

/CECI/soft
Used only by the sysadmins for software installations

CECI Common storage

external remote storage accesible by all clusters

For more details check our detailed documentation
https://support.ceci-hpc.be/doc/_contents/ManagingFiles/TheCommonFilesystem.html



Used space and quotas?

Just use the `ceci-quota` command on any cluster

```
 [alozano@dragon2.dragon2-ctrl0: ~]---> $ ceci-quota

 Diskquotas for user alozano
Filesystem        used      limit      files      limit
$HOME          7.3 GiB   40.0 GiB     205641  unlimited
$CECIHOME     11.4 GiB  100.0 GiB       4390     100000
$CECITRSF     64.0 kiB    1.0 TiB          8  unlimited
```

```
 [alozano@lemaitre3.lm3-w001: ~]---> $ ceci-quota

 Diskquotas for user alozano
Filesystem        used      limit      files      limit
$HOME            4.14G       100G      3.82K
/scratch        4.3 GB  unlimited          8  unlimited
$CECIHOME     11.4 GiB  100.0 GiB       4390     100000
$CECITRSF     64.0 kiB    1.0 TiB          8  unlimited
```

Jobs submission

Batch scripts are submitted and handled by Slurm
How do we control the data on the different storage locations in a job?

There are several variables defined on the job environment, relevants:

`$SLURM_JOB_ID` the Job ID value
`$SLURM_SUBMIT_DIR` directory where the job was submitted from

There are extra environment variables defined by us to point to storage:

`$HOME`

`$LOCALSCRATCH`

`$GLOBALSCRATCH`

`$CECIHOME`

Example of basic sequential write

```
#!/bin/bash
#SBATCH --job-name=job-test
#SBATCH --time=00:15:00 # hh:mm:ss
#SBATCH --ntasks=1
#SBATCH --mem-per-cpu=2000 # megabytes
#SBATCH --partition=batch

echo ""
hn=`hostname`
echo "running on $CLUSTER_NAME node: $hn"

echo ""
echo dump file to GLOBALSCRATCH: $GLOBALSCRATCH

dd if=/dev/zero of=$GLOBALSCRATCH/testdump bs=1M count=2048
sync

echo ""
echo dump file to LOCALSCRATCH: $LOCALSCRATCH

dd if=/dev/zero of=$LOCALSCRATCH/testdump bs=1M count=2048
sync

echo ""
echo dump file to HOME: $HOME

dd if=/dev/zero of=$HOME/testdump bs=1M count=2048
sync

echo ""
echo dump file to CECIHOME: $CECIHOME

dd if=/dev/zero of=$CECIHOME/testdump_lm3 bs=1M count=2048
sync
```

Please **DON'T** run this on your own, is just for illustrative purposes !!

Example of basic sequential write

```
running on lemaitre3 node: lm3-w080.cluster


dump file to GLOBALSCRATCH: /scratch/ulb/operations/alozano
2048+0 records in
2048+0 records out
2147483648 bytes (2.1 GB) copied, 1.66903 s, 1.3 GB/s

dump file to LOCALSCRATCH: /localscratch/alozano/69260406
2048+0 records in
2048+0 records out
2147483648 bytes (2.1 GB) copied, 1.99117 s, 1.1 GB/s

dump file to HOME: /home/ulb/operations/alozano
2048+0 records in
2048+0 records out
2147483648 bytes (2.1 GB) copied, 5.33424 s, 403 MB/s

dump file to CECIHOME: /CECI/home/ulb/operations/alozano
2048+0 records in
2048+0 records out
2147483648 bytes (2.1 GB) copied, 18.8179 s, 114 MB/s
```

Similar order of magnitudes for both *SCRATCH

In the case of multithreaded multinode jobs GLOBALSCRATCH performance can be pushed higher (and is the only option anyway for those jobs)

An order of magnitude below respect the others

Example of basic sequential write

```
running on hercules node: her2-w113

dump file to GLOBALSCRATCH: /workdir/alozano
2048+0 records in
2048+0 records out
2147483648 bytes (2.1 GB) copied, 5.24254 s, 410 MB/s

dump file to LOCALSCRATCH: /scratch/202120023
2048+0 records in
2048+0 records out
2147483648 bytes (2.1 GB) copied, 1.19075 s, 1.8 GB/s

dump file to HOME: /home/alozano
2048+0 records in
2048+0 records out
2147483648 bytes (2.1 GB) copied, 9.93967 s, 216 MB/s

dump file to CECIHOME: /CECI/home/ulb/operations/alozano
2048+0 records in
2048+0 records out
2147483648 bytes (2.1 GB) copied, 13.4418 s, 160 MB/s
```

LOCALSCRATCH is an order of magnitude above all other solutions

But still GLOBALSCRATCH is there to be used (or to store data after a job is done with I/O LOCALSCRATCH)

These are still lower than the others

Example of basic sequential write

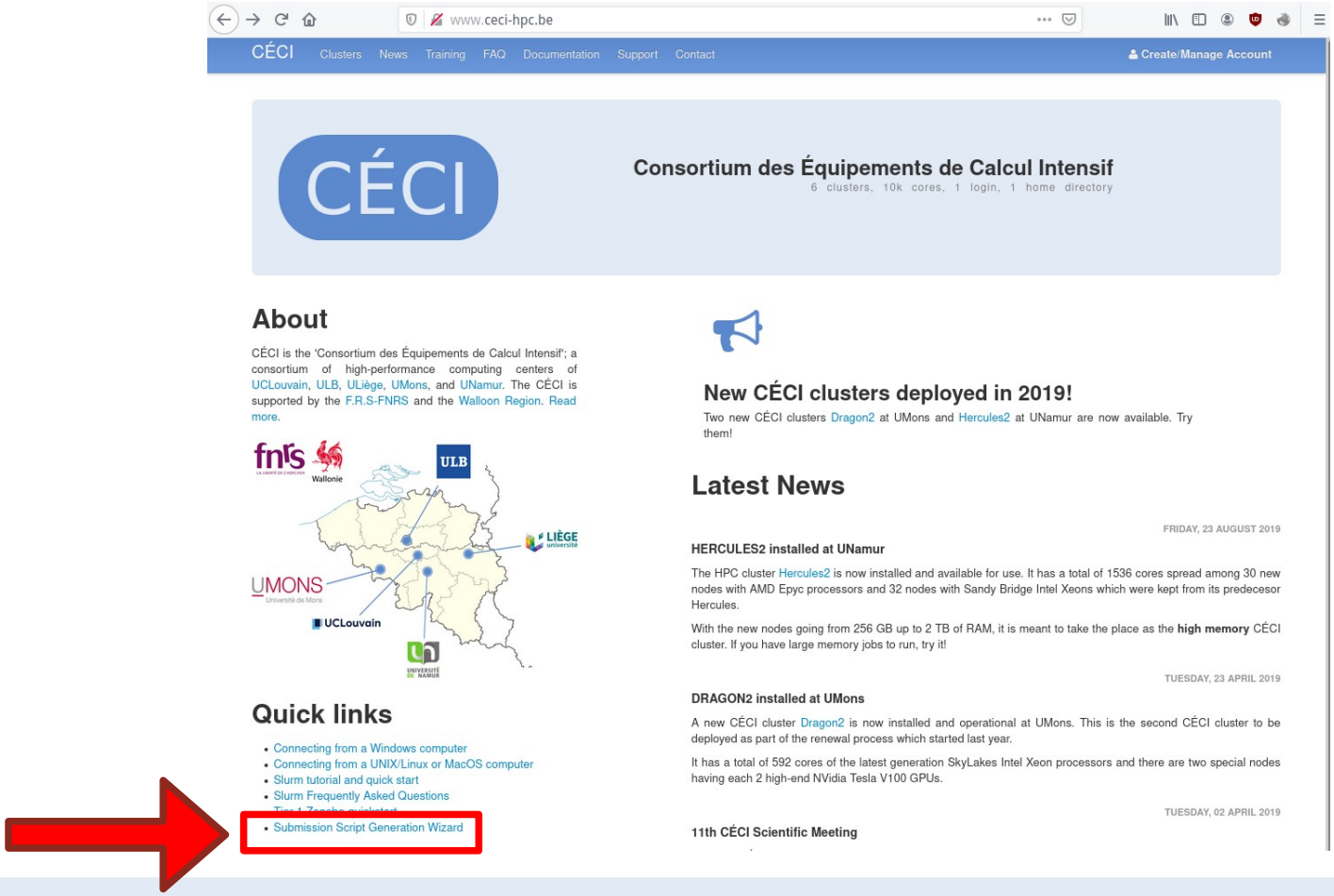
```
running on dragon2 node: drg2-w017

dump file to GLOBALSCRATCH: /globalscratch
2048+0 records in
2048+0 records out
2147483648 bytes (2.1 GB) copied, 2.49736 s, 860 MB/s

dump file to LOCALSCRATCH: /scratch/
2048+0 records in
2048+0 records out
2147483648 bytes (2.1 GB) copied, 0.959233 s, 2.2 GB/s

dump file to HOME: /home/ulb/operations/alozano
2048+0 records in
2048+0 records out
2147483648 bytes (2.1 GB) copied, 3.89209 s, 552 MB/s

dump file to CECIHOME: /CECI/home/ulb/operations/alozano
2048+0 records in
2048+0 records out
2147483648 bytes (2.1 GB) copied, 3.12246 s, 488 MB/s
```

LOCALSCRATCH is the most performant solution

But still GLOBALSCRATCH is there to be used (or to store data after a job is done with I/O on LOCALSCRATCH)

These are still lower than the others

Jobs submission



Jobs submission

www.ceci-hpc.be/scriptgen.html

CÉCI Clusters News Training FAQ Documentation Support Contact Create/Manage Account

Warning: this is still beta. Please send feedback to damien.francois@uclouvain.be. Reload the page to reset.

1. Describe your job

Email address: user@example.com

Job name: Some name

Project: Some project

Output file: (default)

Parallelization paradigm(s)

- Embarrassingly parallel / Job array
- Shared memory / OpenMP
- Message passing / MPI
- GPU / CUDA

Job resources

Duration : 0 days, 1 hour, 0 minutes.

Memory : 2625 MB

Filesystem

Filesystem: $LOCALSCRATCH

Total CPUs: 1 | Total Memory: 2625 MB | Total CPU.Hours: 1

2. Choose a cluster

- NIC4
- Vega
- Lemaitre3
- Hercules2
- Dragon1
- Dragon2
- Zenobe*

3. Copy-paste your script

```
#!/bin/bash
# Submission script for Lemaitre3
#SBATCH --time=01:00:00 # hh:mm:ss
#
#SBATCH --ntasks=1
#SBATCH --mem-per-cpu=2625 # megabytes
#SBATCH --partition=batch,debug

mkdir -p "$LOCALSCRATCH/$SLURM_JOB_ID"
cp -r "$SLURM_SUBMIT_DIR/{your_code,your_input_data}"
"$LOCALSCRATCH/$SLURM_JOB_ID"

cp -r "$LOCALSCRATCH/$SLURM_JOB_ID/your_ouput_data" "$SLURM_SUBMIT_DIR/" &&\
rm -rf "$LOCALSCRATCH/$SLURM_JOB_ID"
```

Examples ….

We are going to check the examples available on the clusters at:

`/CECI/proj/training/ceci_storages`

To wrap up

- Lemaitre3 (and soon NIC5)
You can always rely on using $GLOBALSCRATCH but feel free to profit of $LOCALSCRATCH as is there, if your jobs are single node and data can fit there

- Hercules, Dragon2, Dragon1
If your jobs allow it **prioritize** the usage of $LOCALSCRATCH
But remember this area is shared with other users and there's no quota!!

- **Never** redirect outputs to -> /tmp use always $LOCALSCRATCH instead

IMPORTANT: No data has any kind of backups, neither on the CECI clusters nor Common Storage !!!
Please organize yourself to copy from time to time your important data to some external solution that you own or have access

Thanks for listening!