



Consortium des Équipements de Calcul Intensif

Efficient data storage on the CECI clusters

Ariel Lozano

CÉCI HPC Training
27 Nov 2024

DISCLOSURE



WARNING: No data on the CECI clusters has backups

You are responsible of copying over your useful data you need to store long term somewhere else

Some of the CECI universities provide solutions see:

https://support.ceci-hpc.be/doc/_contents/ManagingFiles/LongtermStorage.html

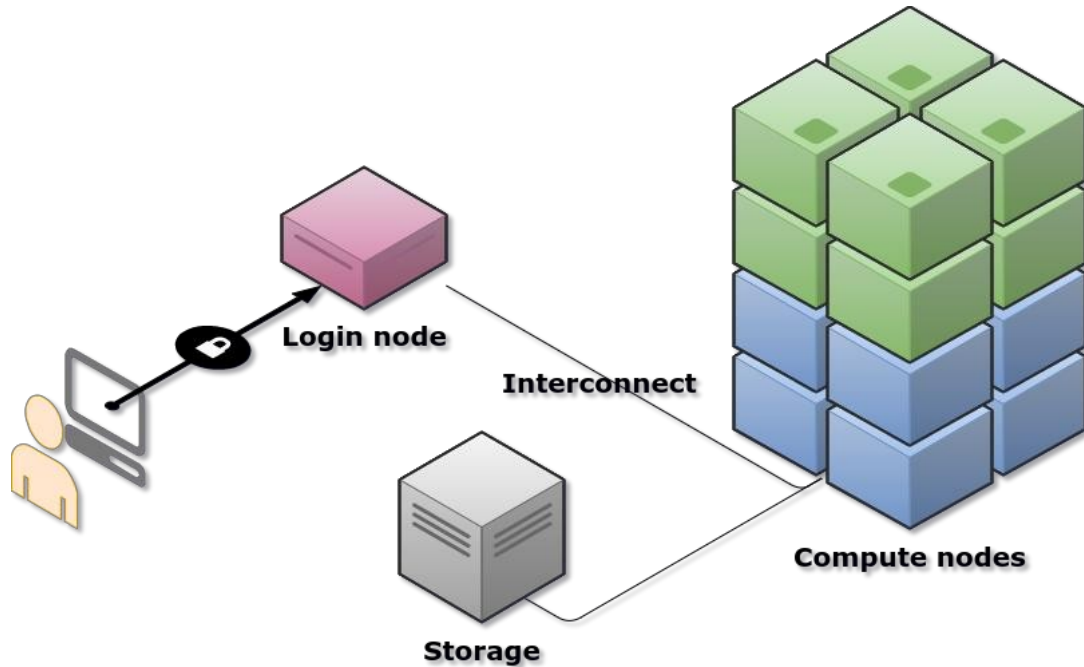
Some context

- Nowadays the best performant **units** of long term storage provides ~2 GB/s of sequential read/write. This can go down to about ~400MB/s for random read/write of many small files.
- Basic sequential write test on a laptop with a consumer NVMe SSD: *2TB Intel SSD 660P Series*

```
$ 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, 0.842955 s, 2.5 GB/s
```

- Basic test with a single task writing on the storage.
- The CPU access the SSD directly via PCI express lanes.

Previous: HPC cluster



"Introduction to high-performance computing" (Frédéric Wautelet)

- A computer 'cluster' is a group of **linked** computers working together closely, so that in many respects they form a single computer
- Corollary: Access to **most** of the different storage solutions happens via the network

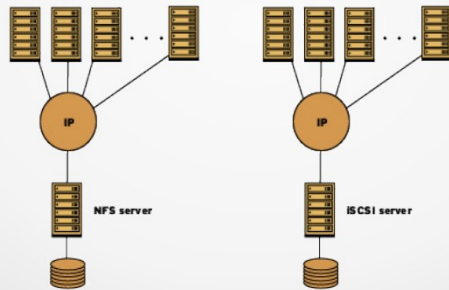
Previous: Network storage solutions

Network filesystem



One source many consumers

NAS: ex. NFS SAN: ex. GFS2



Typical usage: Home directories, Mass storage

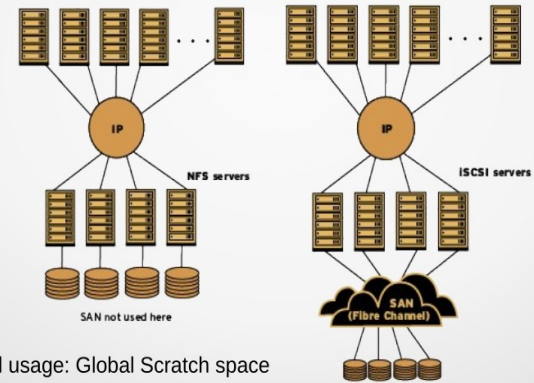
10

Pictures from https://www.redhat.com/magazine/008jun05/features/gfs_nfs/

Parallel / distributed filesystem



Many sources many consumers
ex: Lustre, GPFS, BeeGeeFS GlusterFS



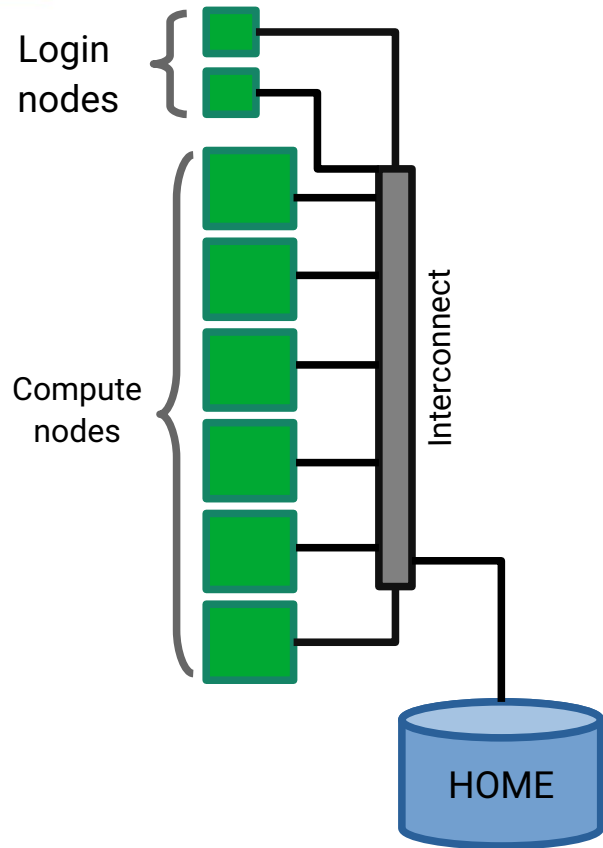
Typical usage: Global Scratch space

11

Pictures from https://www.redhat.com/magazine/008jun05/features/gfs_nfs/

Damien François, "Introduction to data storage and access"

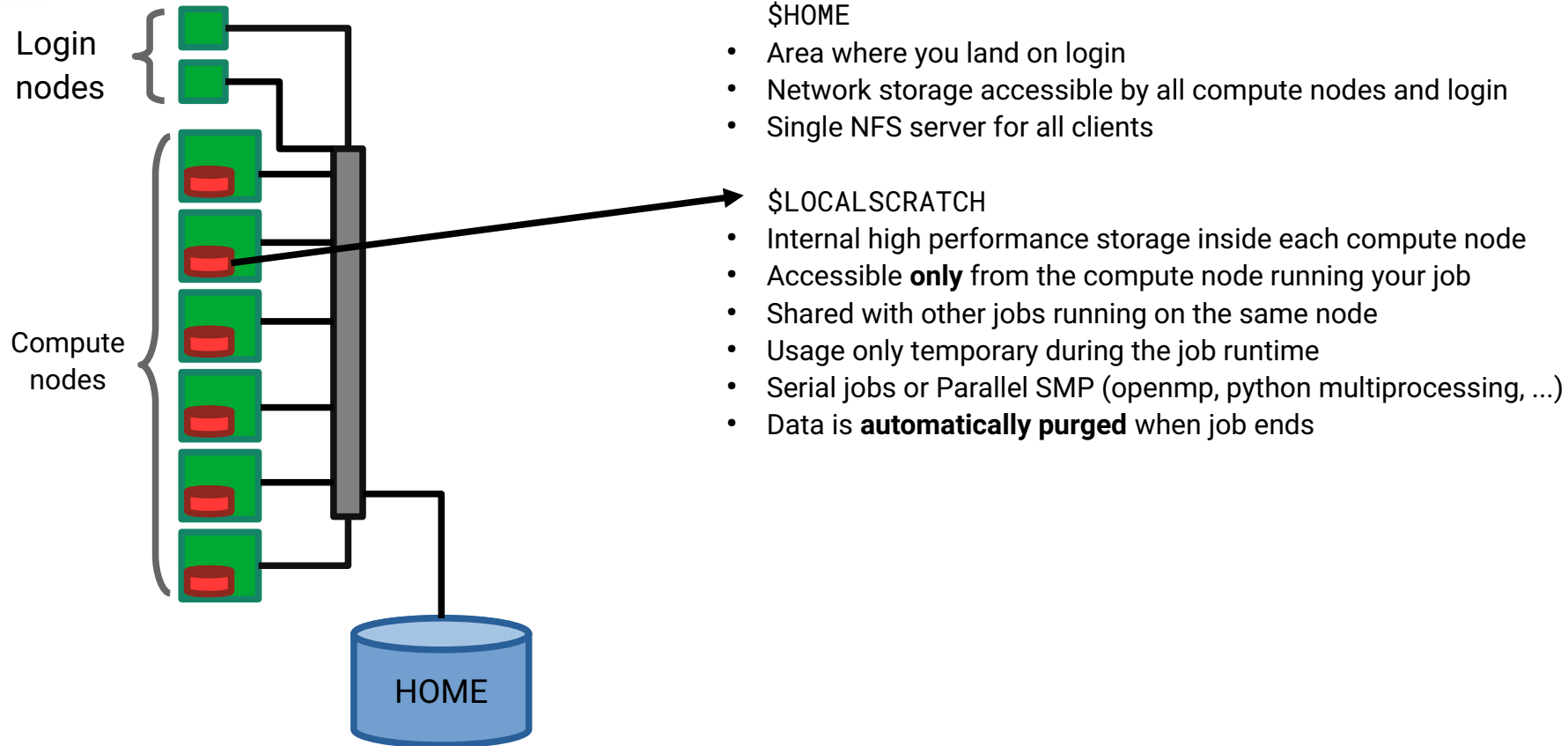
Storages on CECI clusters



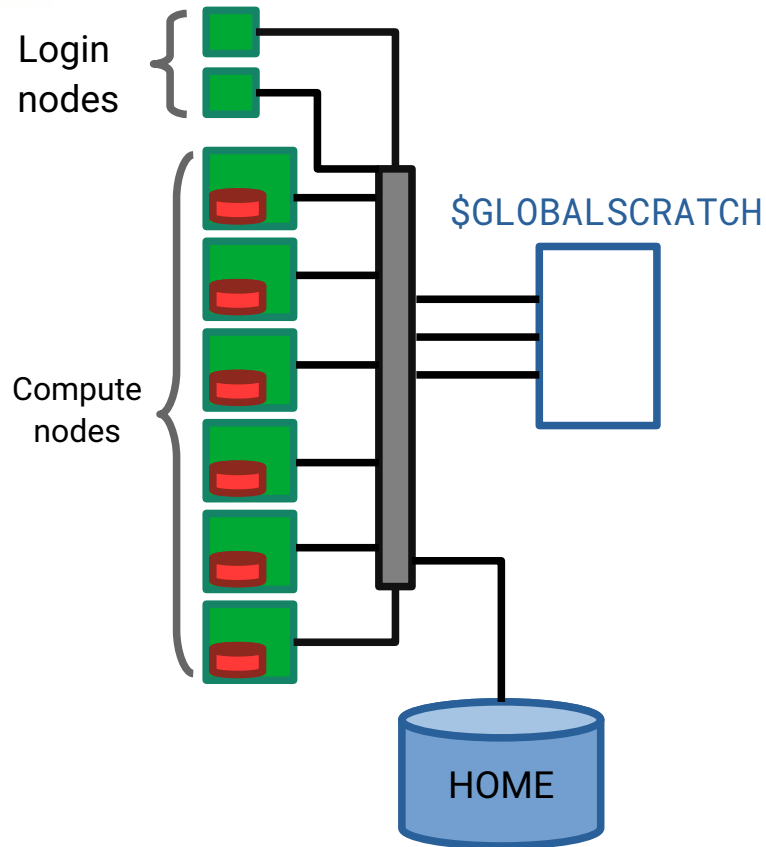
\$HOME

- Area where you land on login
- Network storage accessible by all compute nodes and login
- Single NFS server for all clients

Storages on CECI clusters



Storages on CECI clusters



\$HOME

- Area where you land on login
- Network storage accessible by all compute nodes and login
- Single NFS server for all clients

\$LOCALSCRATCH

- Internal high performance storage inside each compute node
- Accessible **only** from the compute node running your job
- Shared with other jobs running on the same node
- Usage only temporary during the job runtime
- Serial jobs or Parallel SMP (openmp, python multiprocessing, ...)
- Data is **automatically purged** when job ends

\$GLOBALSCRATCH

- Implemented via different setups but usually a parallel filesystem
- Network storage accessible by all compute nodes and login
- Can be composed of a single or multiple storage sources
- All jobs but **only option** for **multinode-parallel** jobs (big MPI jobs)
- Data there stays persistently (but all is removed in yearly maintenances)
- Please cleanup from time to time

Storages on CECI clusters

- How much data can we handle on each area? check the `ceci-quota` command

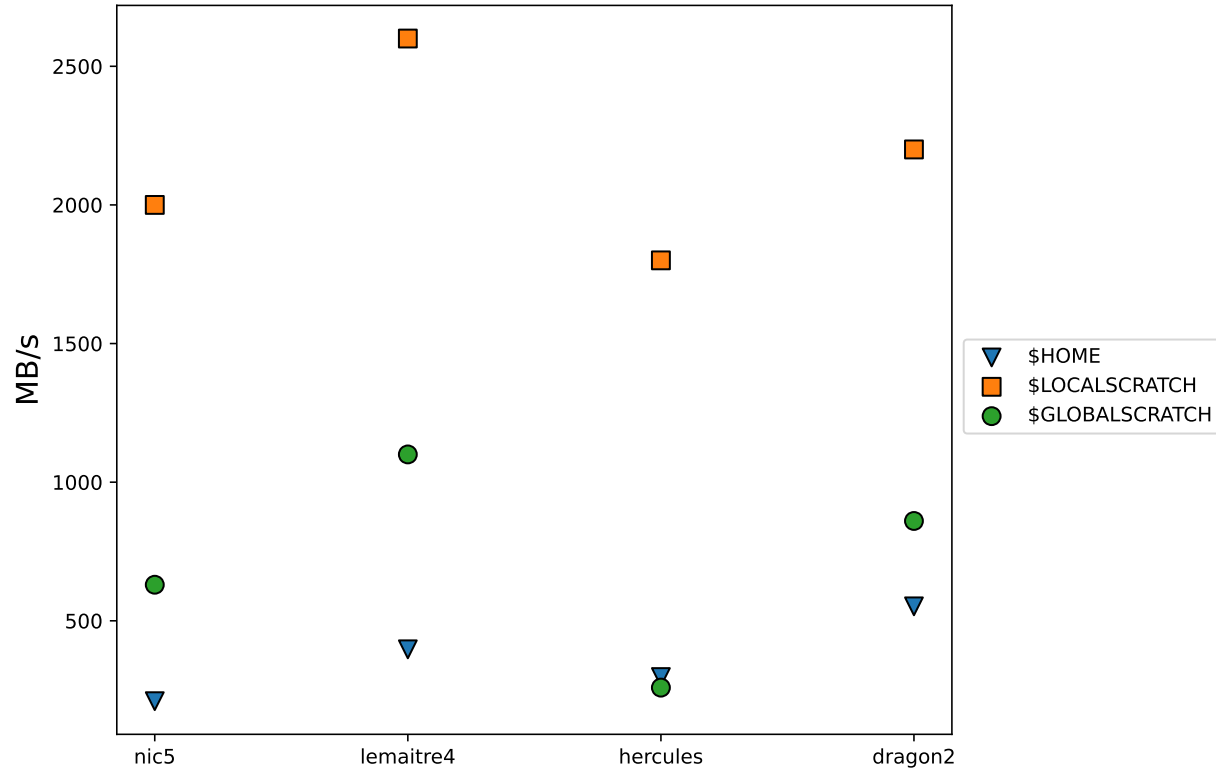
Cluster	\$HOME (nfs server)	\$LOCALSCRATCH (hardware support)	\$GLOBALSCRATCH (fstype)
NIC5	100 GB	370GB (local SSD)	5TB (520TB beegfs)
Lemaitre4	100 GB	180GB (local SSD)	XX ^[1] (320TB beegfs)
Dragon2	40 GB	3TB (HDD RAID0)	XX ^[1] (60TB beegfs)
Hercules	200 GB	her2-w065..096: 1TB her2-w099..126: 4TB her2-w127..128: 8TB (HDD RAID0)	400GB soft / 4TB hard (nfs server)

More info: https://support.ceci-hpc.be/doc/_contents/ManagingFiles/Storage.html#quota-label

^[1]No limits enforced but remember resources are shared among all ~800 CECI users!!

Storages on CECI clusters

Sequential write speed (dump of a 2GB file)



- This is **NOT** a definitive benchmark of the filesystems. Pattern is of a single core job doing a single write at the end.
- Different I/O patterns on your job will produce diverse results

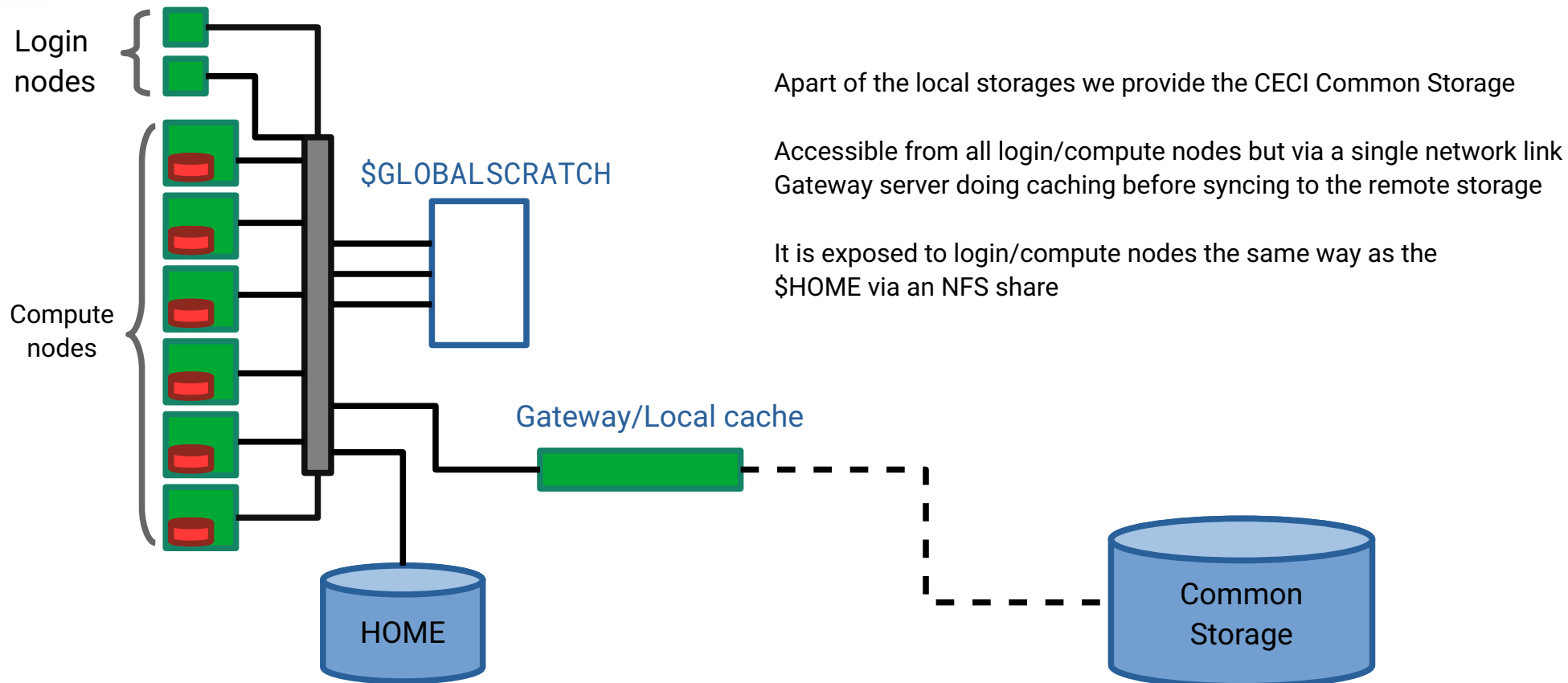
Storages on CECI clusters

Wrap-up

- `$LOCALSCRATCH` will be always the ideal for single core jobs
- If all your input/output data volume for your job fits on `$LOCALSCRATCH` **use it** your code will speed up just for doing that
- `$GLOBALSCRATCH` is the only choice for parallel multi-node jobs
- `$GLOBALSCRATCH` can also be used to store big input data, or recover useful output for a job working on `$LOCALSCRATCH`
- There's **no** reason to use `$HOME` for job I/O. Performance is the **lowest** and the usable space the **smallest**

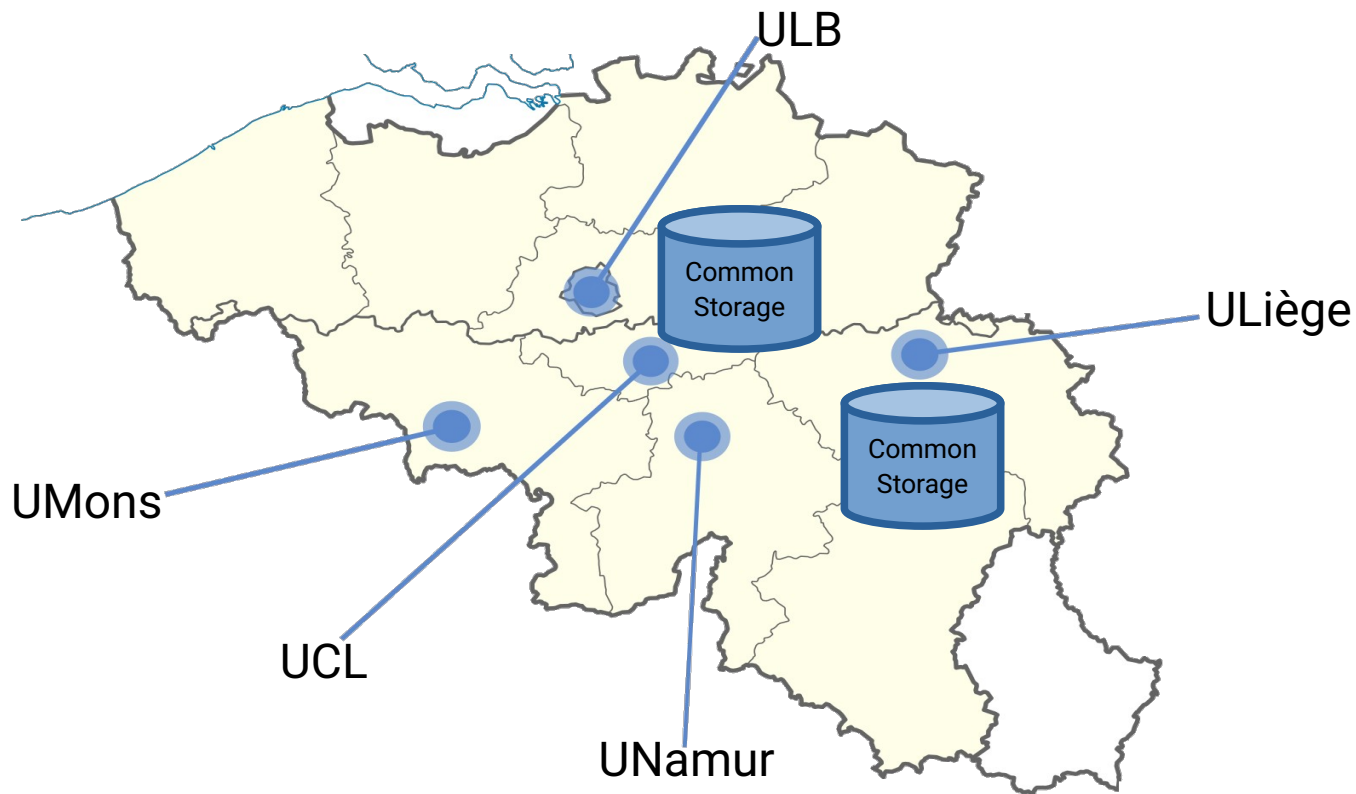
CECI Common storage

external remote storage accessible by all clusters



CECI Common storage

external remote storage accesible by all clusters



The main storage servers are in ULiège and UCL

There is a dedicated data interconnect among the 5 sites for this solution

CECI Common storage

external remote storage accesible by all clusters

/CECI/home

- Personal directory path defined via \$CECIHOME variable
- Quota of 100GB

/CECI/trsf

- Personal directory path defined via \$CECITRSF variable
- Meant only for **temporary** copying from one cluster to another
- Data here is subject to be purged every 6 months
- Quota of 1TB soft 10TB hard

/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/soft

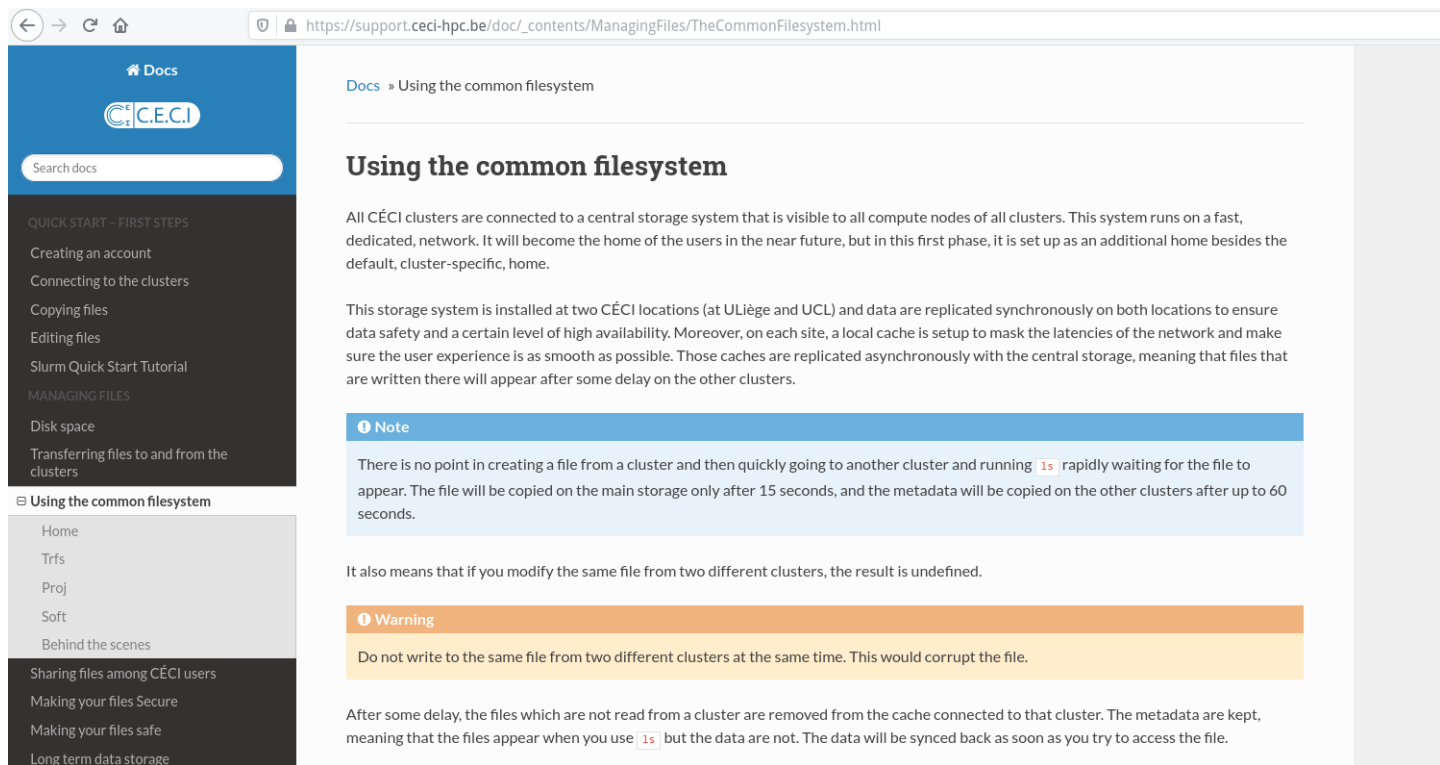
Used only by the sysadmins for software installations

CÉCI Common storage

external remote storage accesible by all clusters

For more details check our detailed documentation

https://support.cec-hpc.be/doc/_contents/ManagingFiles/TheCommonFilesystem.html



The screenshot shows a web browser displaying the CÉCI documentation page. The browser's address bar shows the URL: https://support.cec-hpc.be/doc/_contents/ManagingFiles/TheCommonFilesystem.html. The page has a blue header with the CÉCI logo and a search bar. A left sidebar contains a navigation menu with categories like 'QUICK START - FIRST STEPS', 'MANAGING FILES', and 'Using the common filesystem'. The main content area is titled 'Using the common filesystem' and includes a breadcrumb 'Docs » Using the common filesystem'. The text explains that all CÉCI clusters are connected to a central storage system. A blue 'Note' box states that there is a 15-second delay for file replication. An orange 'Warning' box advises against writing to the same file from two different clusters simultaneously. The page concludes by noting that files not read from a cluster are removed from its cache, but metadata is kept.

Docs » Using the common filesystem

Using the common filesystem

All CÉCI clusters are connected to a central storage system that is visible to all compute nodes of all clusters. This system runs on a fast, dedicated, network. It will become the home of the users in the near future, but in this first phase, it is set up as an additional home besides the default, cluster-specific, home.

This storage system is installed at two CÉCI locations (at ULiège and UCL) and data are replicated synchronously on both locations to ensure data safety and a certain level of high availability. Moreover, on each site, a local cache is setup to mask the latencies of the network and make sure the user experience is as smooth as possible. Those caches are replicated asynchronously with the central storage, meaning that files that are written there will appear after some delay on the other clusters.

Note

There is no point in creating a file from a cluster and then quickly going to another cluster and running `ls` rapidly waiting for the file to appear. The file will be copied on the main storage only after 15 seconds, and the metadata will be copied on the other clusters after up to 60 seconds.

It also means that if you modify the same file from two different clusters, the result is undefined.

Warning

Do not write to the same file from two different clusters at the same time. This would corrupt the file.

After some delay, the files which are not read from a cluster are removed from the cache connected to that cluster. The metadata are kept, meaning that the files appear when you use `ls` but the data are not. The data will be synced back as soon as you try to access the file.

Used space and quotas?

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

```
[myuser@dragon2.dragon2-ctrl10: ~]---> $ ceci-quota
```

```
Diskquotas for user myuser
```

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

```
[myuser@lm4-f001: ~]---> $ ceci-quota
```

```
Diskquotas for user myuser
```

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

How do we control the data location from a Slurm job?

Making use of the pre-defined environment variables:

`$HOME`

`$LOCALSCRATCH`

`$GLOBALSCRATCH`

`$CECIHOME`

Extra useful variables defined on-the-fly when submitting a job:

`$SLURM_JOB_ID` the Job ID value

`$SLURM_SUBMIT_DIR` directory where the job was submitted from

Jobs submission

LOCALSCRATCH example

```
#!/bin/bash
#SBATCH --job-name=stest1
#SBATCH --time=00:15:00
#SBATCH --ntasks=1
#SBATCH --mem-per-cpu=1000
#SBATCH --partition=batch

module load releases/2021b
module load SciPy-bundle/2021.10-foss-2021b

# define some useful directory names
RUNDIR="${LOCALSCRATCH}"
SUBMITDIR="${SLURM_SUBMIT_DIR}"

# on this example we assume our input files are on the
# same directory where we run sbatch myjob.sh
cp -r ${SUBMITDIR}/input_1000.in ${RUNDIR}/input.in

# change to the run directory on the compute node
cd ${RUNDIR}

# execute your code
python ${HOME}/project1/mycode/mycode1.py

# recover your useful output
cp -r long.out ${SUBMITDIR}/
```

- Python code that requires just an input file “input.in”
- The code is located on a folder inside my \$HOME
- The only useful output it will produce is “long.out”

Jobs submission

GLOBALSCRATCH example

```
$ mkdir -p $GLOBALSCRATCH/myjob1000
$ cp -r input_1000.in $GLOBALSCRATCH/myjob1000/input.in
$ cd $GLOBALSCRATCH/myjob1000

> i create my submission script myjob.sh as e.g. below

#!/bin/bash
#SBATCH --job-name=gtest1
#SBATCH --time=00:15:00
#SBATCH --ntasks=1
#SBATCH --mem-per-cpu=1000
#SBATCH --partition=batch

module load releases/2021b
module load SciPy-bundle/2021.10-foss-2021b

python ${HOME}/project1/mycode/mycode1.py

> and we can submit it directly from here
$ sbatch myjob.sh
```

We can prepare everything before inside \$GLOBALSCRATCH, input and submission script, then submit from there without extra data movements.

This is just an illustrative example !

For single node jobs where all input/output fits on LOCALSCRATCH use the previous approach

Or also **consider a mix** if the input data is too big, but the produced output fits on LOCALSCRATCH

Jobs submission

The screenshot shows the CÉCI website interface. At the top, there is a navigation bar with links for Clusters, News, Training, FAQ, Documentation, Support, and Contact, along with a 'Create/Manage Account' button. Below this is a large header area with the CÉCI logo and the text 'Consortium des Équipements de Calcul Intensif' and '6 clusters, 10k cores, 1 login, 1 home directory'. The main content area is divided into two columns. The left column has an 'About' section with a map of Belgium showing the locations of member universities: ULB, LIÈGE université, UMONS, and UCLouvain. Below this is a 'Quick links' section with a list of links, including 'Submission Script Generation Wizard' which is highlighted with a red box and a red arrow. The right column features a 'New CÉCI clusters deployed in 2019!' announcement with a megaphone icon, followed by a 'Latest News' section with three articles: 'HERCULES2 installed at UNamur', 'DRAGON2 installed at UMONS', and '11th CÉCI Scientific Meeting'.

About

CÉCI is the 'Consortium des Équipements de Calcul Intensif'; a consortium of high-performance computing centers of UCLouvain, ULB, ULiège, UMONS, and UNamur. The CÉCI is supported by the F.R.S-FNRS and the Walloon Region. [Read more.](#)

Quick links

- [Connecting from a Windows computer](#)
- [Connecting from a UNIX/Linux or MacOS computer](#)
- [Slurm tutorial and quick start](#)
- [Slurm Frequently Asked Questions](#)
- [The 11th Scientific Meeting](#)
- [Submission Script Generation Wizard](#)

New CÉCI clusters deployed in 2019!

Two new CÉCI clusters [Dragon2](#) at UMONS and [Hercules2](#) at UNamur are now available. Try them!

Latest News

HERCULES2 installed at UNamur FRIDAY, 23 AUGUST 2019

The HPC cluster [Hercules2](#) is now installed and available for use. It has a total of 1536 cores spread among 30 new nodes with AMD Epyc processors and 32 nodes with Sandy Bridge Intel Xeon which were kept from its predecessor Hercules.

With the new nodes going from 256 GB up to 2 TB of RAM, it is meant to take the place as the **high memory** CÉCI cluster. If you have large memory jobs to run, try it!

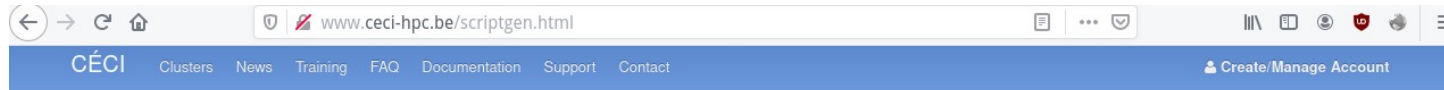
DRAGON2 installed at UMONS TUESDAY, 23 APRIL 2019

A new CÉCI cluster [Dragon2](#) is now installed and operational at UMONS. This is the second CÉCI cluster to be deployed as part of the renewal process which started last year.

It has a total of 592 cores of the latest generation SkyLakes Intel Xeon processors and there are two special nodes having each 2 high-end NVidia Tesla V100 GPUs.

11th CÉCI Scientific Meeting TUESDAY, 02 APRIL 2019

Jobs submission



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

1. Describe your job

Email address:

Job name:

Project:

Output file:

Parallelization paradigm(s)

Embarrassingly parallel / Job array

Shared memory / OpenMP

Message passing / MPI

GPU / CUDA

Job resources

Duration : days, hour, minutes.

Memory : MB

Filesystem

Filesystem:

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_output_data" "$SLURM_SUBMIT_DIR/" &&
rm -rf "$LOCALSCRATCH/$SLURM_JOB_ID"
```



Examples

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

```
cat /CECI/proj/training/ceci_storages/README.md
```

To wrap up

- For all clusters

Never do direct I/O on your \$HOME

Prioritize the usage of \$LOCALSCRATCH if your jobs allow it (e.g. jobs running on a single node)
Remember this area is shared with other users of the node and there's no quota!!

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

- Lemaitre4 and NIC5

For your multi-node parallel jobs **always** use \$GLOBALSCRATCH **never** your \$HOME



Remember to backup your useful data somewhere else outside the clusters