Table of Contents

- Storage
- Cluster storage
  - Home directory
    - Quota
  - Scratch Directory
    - Quota
    - Data Retention Policy
  - Fast directory
    - Quota
- Local storage
  - Scratch directory (local on each node)
  - Temporary private space
  - Temporary shared space
- Sharing files with other users
- Best practices
  - I/O performance
  - Check disk usage on the clusters
    - Check disk usage on home and scratch
    - Check disk usage on NASAC
  - File transfer (between nodes)
- Backup
- Archive
- Access external storage
  - NASAC
    - Troubleshooting
      - Where does gio mounts my data?
      - List the user DBUS process
      - Sometimes mount is not available but you can browse/copy/interract with gio commands
  - CVMFS
  - EOS
- Robinhood
    - Policies
    - Report

Storage

There are different types of storage on the clusters. This is important to understand where to store which type of data.

Important: please note that Boabab and Yggdrasil have both the same architecture and logic regarding storage, however the data is **not shared** between the clusters.

Cluster storage

We use a distributed parallel filesystem on our clusters called *BeeGFS*. Any data you put in `$HOME` or in `$HOME/scratch` folders are thereby accessible on the login node and on each compute node of the cluster. This means you don't need to waste time or resources to copy data on the nodes.

This is the storage space we offer on our clusters

Cluster	Path	Total storage size	Nb of servers	Nb of targets per servers	Backup	Quota size	Quota number files
Baobab	/home/	138 TB	4	1 meta, 2 storage	Yes (tape)	1 TB	-
	/srv/beegfs/scratch/	1.0 PB	2	1 meta, 6 storage	No	-	10 M
	/srv/fast	5 TB	1	1	No	500G/User 1T/Group	-
Yggdrasil	/home/	495 TB	2	1 meta, 2 storage	Yes (tape)	1 TB	-
	/srv/beegfs/scratch/	1.2 PB	2	1 meta, 6 storage	No	-	10 M

We realize you all have different needs in terms of storage. To guarantee storage space for all users, we have **set a quota on home and scratch directory**, see table above for details. Beyond this limit, you will not able to write to this filesystem. We count on all of you to only store research data on the clusters. We also count on your **to periodically delete old or unneeded files** and to **clean up everything when you will leave UNIGE**. Please keep on reading to understand when you should use each type of storage.

Home directory

Your home folder is located here: `$HOME`.
It is available on the login node and on each compute nodes of the cluster.

In your home directory you can store any file needed for running your jobs : data, code, software, etc.

However, this is **not a personal storage** for data such as emails, private pictures, etc.

Please always use the the `$HOME` variable in your script instead of using the full path. This is a good practice and will save you a lot of trouble.

A backup of your `$HOME` folder is done daily.

For temporary storage, please read about the “Scratch directory” below.

Your `$HOME` is only accessible by you (permission `0700`) ; you are not allowed to change the permissions and if you do, they are automatically reset every day. If you need to share files, please check Sharing files with other users.

Quota

As the storage is shared by everyone, this ensure a fair home usage and prevent users from filling it. It is important to lower the disk usage on home as this location is backuped and this takes a lot of time, especially if you have a lot of files.

What does it mean for you: if your home directory usage is higher than 1TB, you won't be able to write to it anymore.

Error message:

```
Disk quota exceeded
```

To resume the situation, you should clean up some data in your home directory and or migrate some data to your scratch directory.

Scratch Directory

Location and Accessibility:

Your scratch directory is located at: `$HOME/scratch`.

- It is available on both the login node and all compute nodes of the cluster.
- It offers more storage space than `$HOME`.
- It is not backed up

N.B.: `$HOME/scratch` is a symbolic link to your personal folder in `/srv/beegfs/scratch/`.

Purpose of the Scratch Directory:

The scratch directory is intended for storing non-unique or regenerable data. You should use it for:

- Temporary storage of application-generated files.
- Large datasets used as input during computations.
- However, this directory is not backed up. Please avoid storing critical data here.

Permissions and Access Control:

- Your `$HOME/scratch` is **only** accessible by you (permission `0700`).
- Permission modifications are not allowed and will be **automatically** reset.
- If you need to share files, refer to: Sharing files with other users.

Best Practices:

The scratch directory is **not a permanent** storage solution. To ensure efficient use:

- Regularly clean up unnecessary files.
- Move important data elsewhere once your project is completed. Unige provides Storage solution which can be mounted on Clusters

Quota

Since the scratch storage is shared among all users, a file count quota is enforced to ensure fair usage:

- Maximum file count: 10 million (10M)

If you exceed this limit, you won't be able to write any new files.

Data Retention Policy

Important: The data retention policy will be implemented on Baobab during the next maintenance from February 18 to 21, 2025. More Info

Automatic Deletion Rules:

- Files **older than 3 months** will be automatically deleted.
- Deletion is based on the last access (read or writte) date of each file.

What This Means for You:

- Any file not accessed within the last 3 months will be considered inactive and deleted.
- Frequently used files will remain unaffected.

By following these guidelines, you can ensure efficient use of the scratch storage while avoiding data loss.

Fast directory

A new fast storage is available dedicated for jobs using multiples nodes and scratchlocal need to be shared between nodes.

Cluster	path	Total storage size	Nb of servers	Backup	Quota size	Quota number files
Baobab	`/srv/fast`	5 TB	1	No	500G by user & 1TB by group	-

This storage will be erased on each maintenances.

Quota

As the storage is shared by everyone, this ensure a fair scratch usage and prevent users from filling it. We setup a quota based on the total size.

You should clean up some data in your fast directory as soon as your jobs are finished.

Local storage

Scratch directory (local on each node)

On **each** compute node, a local `scratch` folder is available and is located here : `/scratch`.
It is available on all compute node of the cluster. Unlike `$HOME/scratch`, the local scratch can only be used **from the node** itself and **while** your job is running.

It is usually fast and there is no overhead for using the network. It is also more efficient at dealing with a large amount of small files.

You usually use it for temporary files generated while your job is running and that do not need to be accessible at the end of the job.

Important : at the end of the computation, all files in that directory are **automatically deleted**. If you need to keep some files, you can add a command to your Slurm sbatch script to move them in the `$HOME` or `$HOME/scratch` directory at the end of the job.

While all the Yggdrasil nodes have local SSDs, on Baobab we still have HDDs (*i.e.* mechanical disks) on the following nodes:
`node[001-005,007-021,023-056,058-059,061-066,093-101,103-111,140-156,180,186,203]`.

Temporary private space

On **each** compute node, you can use the following private ephemeral spaces:

- `/dev/shm`: fastest storage, on RAM, be careful to request enough RAM)
- `/var/tmp`: on local disk
- `/tmp`: on local disk

Those places are private and only accessible by your job.

Temporary shared space

If you need to access the data from more than one node, you can use a space reachable from all your jobs running on the same compute node. When you have no more jobs running on the node, the content of the storage is erased.

The path is the following: `/share/users/${SLURM_JOB_USER:0:1}/${SLURM_JOB_USER}`

See here for a usage example:
https://hpc-community.unige.ch/t/local-share-directory-beetween-jobs-on-compute/2893

Sharing files with other users

Sometimes, you may need to share files with colleagues or members of your research group.

We offer two types of shared folders:

- **In the “home” directory** (`/home/share/`): Ideal for sharing scripts and common libraries related to a project.
- **In the “scratch” directory** (`/srv/beegfs/scratch/shares/`): Suitable for sharing larger files, such as datasets.

To request a shared folder, please fill out the form at DW. As part of the request, you'll be asked if you already have a *group* you’d like to use.

A **group** is a collection of users used to manage shared access to resources. These groups are defined and stored in the **Active Directory** and allow us to control who can access specific folders. If you need more details about groups, please contact your **CI** (*correspondant informatique*).

If you are an *Outsider* user and do not have access to DW, please ask your **PI** to submit the request on your behalf.

You are not allowed to change the permission of your `$HOME/$SCRATCH` folder on the clusters. Even if you did, our automation scripts will break what you did.

For easy sharing you need to set `umask 0002` (thus new files and directories will be created with 660 and 770 permissions, respectively), otherwise you will be asked for confirmation every time you want to modify a file or, even worse, you will not be able to create new files/folders.

This is a side-effect of the default permissions on Red Hat-based systems without **User Private Groups** (*i.e.* when the UID/GID differs, as it is the case on Baobab/Yggdrasil [1]).

Since we use ACL to set the user right, you can't rely on sticky bit to force the new files to belong to a group which is not your primary group. You have the following options:

- You can request to change your primary group: every file that you create on the cluster will belong to this group
- You can set your umask to 0002 as explained previously
- You can launch on a regular basis as script that “fix” the group. Example: `find . -type f -exec chown :share_xxx {} \;`

Best practices

I/O performance

Source : [2)]

All the BeeGFS storage are shared. Although it is a pretty robust solution, the performance depends of the workload of all the users and can be impacted negatively. This is especially true on a cluster, where a single job may perform I/O from many different threads at the same time, even on multiple nodes.

BeeGFS uses two kind of storage servers.

- **metadata** servers: they are involved when you type `ls`, `find`, etc. and you don't read the content. The metadata is stored on fast NVMe disks. See here for details about the infrastructure.
- **storage** servers: they are involved when you read or write a file. The data are stored on regular hard drives disks. See here for details about the infrastructure.

The bottleneck is usually the metadata servers and you notice it while trying to browse files.

A performance killer is for example to read/write a lot of very small files or to traverse a directory with thousands of files. This is usually something the user himself has a control on and can avoid, for example by limiting the number of files per directory.

Also, think twice before launching those tools on a shared storage:

- `updatedb`
- `find`
- `ncdu`
- `du`
- or anything that takes a long time to scan all files on the storage

Best practices :

- limit the number of files per directory (for example, 1000)
- do not test the performance of the storage by launching a "benchmark", as you'll waste precious resources
- work with bigger files (1MB instead of 1KB for example). In this case, size matters and the bigger, the better.
- avoid reading many times the same file, try to cache it instead.

Check disk usage on the clusters

Check disk usage on home and scratch

Since `/home` and `/srv/beegfs/scratch/` have quota enabled and enforced, we can quickly check

the disk usage by fetching the quota information.

The script `beegfs-get-quota-home-scratch.sh` gives you a quick summary :

```
(baobab)-[sagon@login1 ~]$ beegfs-get-quota-home-scratch.sh
home dir: /home/sagon
scratch dir: /srv/beegfs/scratch/users/s/sagon

        user/group                 ||           size          ||    chunk
files
storage     |   name        |  id  ||    used    |    hard    ||  used   |
hard
----------------------------|------||------------|------------||---------|--
-------
home        |          sagon|240477||  530.46 GiB| 1024.00 GiB||
1158225|unlimited
scratch     |          sagon|240477||    2.74 TiB|   unlimited||   436030|
10000000
```

This includes all your data in `$HOME`, `$HOME/scratch`, but also any data in `/home/share` and `/srv/beegfs/scratch/shares` that belongs to you (if you are using a shared directory).

The column “chunk files” doesn't correspond exactly to the number of files you own. It corresponds to the number of chunks you own. Each file has at least one chunk. The current chunk size is 512kB. If a file is bigger, it will be split.

Check disk usage on NASAC

If you have space as well in `/acanas` (The NASAC) you can check your quota and usage like this:

```
-bash-4.2$ quota --hide-device  -s -f /acanas
Disk quotas for user XXX (uid NNN):
     Filesystem   space   quota   limit   grace   files   quota   limit
grace
   1975G      0K   3072G          15725k       0       0
```

File transfer (between nodes)

There is no need to transfer files between compute nodes and the login as they share the same storage space (`$HOME` and `$HOME/scratch`).

However, if for some reason you need specifically to transfer something from the `login` node to the local storage of a compute node, you can use `sbcast` (but you probably shouldn't).

Backup

Your home (`$HOME`) directory is backed up every day.

To avoid backing up temporary data, we ask you to be fair and store them in the `$HOME/scratch/` directory.

There is **no backup** of the `scratch` folder.

If you deleted something from the `$HOME` directory, please contact us by email :

- send the email to hpc@unige.ch
- provide the following :
  - the full path of the folder or file you deleted
  - the date and time (as precise as possible) when you deleted it
  - the date and time (as precise as possible) of the last time you used the file
  - we will try to retrieve the file(s) from the most recent backup available before deletion.

Please be aware that the backup retention is limited. The backup ruls are :

- For “active” files, we keep 10 versions or the ones modified in the last 30 days.
- For “inactive” files (file that are not present on disk, either because they were deleted or moved), we keep the last 2 versions for 90 days. After 90 days, those copies are deleted for good.

Thus we cannot guarantee to recover a version of a file more recent than 24h, or on the contrary a file you deleted a long time ago. **Don't bargain** based on the above mentioned rules, just contact us as soon as possible and we will see what is doable.

Archive

If you need to archive you results, you can use the following services:

- NAS academic, to store archives on a CIFS or NFS share at the price of 75.-CHF / TB / year.
- Yareta for long-term preservation.

Baobab isn't a long term archive service!

Access external storage

If you need to transfer files from “outside” the cluster, please refer to this documentation

NASAC

If you need to mount an external share (NAS for example) on Baobab from command line, you can proceed as follow in your terminal.

Launch dbus:

```
[sagon@login1 ~] $ dbus-launch bash
```

mount the share, smb in this example:

```
[sagon@login1 ~] $ gio mount smb://server_name/share_name
```

This will prompt for the username, password and domain. If you are mounting an UNIGE network share such as the NASAC, it’s your ISIS credentials and the domain is ISIS.

The share will be mounted on `/run/user/your_uid/gvfs/`

You can access the files using standard POSIX tools such as `cp` `ls` etc but keep in mind that those tools aren't meant to be used on non reliable media such as network share. If you face an error when accessing a specific file, you can use `gio copy` or `rsync` as `cp` replacement which works better and handle network errors. The same for `ls` etc.

When you don’t need to access the data anymore, you may unmount the share:

```
[sagon@login1 ~] $ gio mount -u smb://server_name/share_name
```

The data are only available where gio has been mounted. If you need to access the on other nodes, you need to mount them there as well in your sbatch script.

If you need to script this, you can put your credentials in a file in your home directory.

Content example with the credendtials stored in the file `.credentials`.

```
username
domain
password
```

Mount example using credentials in a script:

```
[sagon@login1 ~] $ gio mount smb://server_name/share_name < .credentials
```

Do not store the credentials in the file `.netrc` as the format conflict with other tools using it such as `wget`.

Troubleshooting

Where does gio mounts my data?

The process that lets you access GVfs/Gio shares via CLI is called gvfsd-fuse and its first argument is the folder where GVfs/Gio shares are exposed to.

You can find such folder with the following command:

```
[sagon@login1 ~] $ ps ux | grep -e '[g]vfsd-fuse'
sagon    196919  0.0  0.0 387104  3376 ?        Sl   08:49   0:00
/usr/libexec/gvfsd-fuse /home/sagon/.gvfs -f -o big_writes
```

In this case, it means the mount will be done in `/home/sagon/.gvfs`

List the user DBUS process

```
[sagon@login1 ~] $ pgrep -a -U $(id -u) dbus
196761 /usr/bin/dbus-daemon --fork --print-pid 4 --print-address 6 --session
224317 /usr/bin/dbus-daemon --fork --print-pid 4 --print-address 6 --session
```

reference: [3)]

Sometimes mount is not available but you can browse/copy/interract with gio commands

```
$ dbus-launch bash

$ gio mount smb://nasac-evs2.unige.ch/hpc_exchange/backup
Authentication Required
Enter user and password for share “hpc_exchange” on “nasac-evs2.unige.ch”:
User [rossigng]: s-hpc-share
Domain [SAMBA]: ISIS
Password:

$ gio mount -l
Drive(0): SAMSUNG MZ7L3480HBLT-00A07
  Type: GProxyDrive (GProxyVolumeMonitorUDisks2)
Drive(1): SAMSUNG MZ7L3480HBLT-00A07
  Type: GProxyDrive (GProxyVolumeMonitorUDisks2)
Mount(0): hpc_exchange on nasac-evs2.unige.ch -> smb://nasac-
evs2.unige.ch/hpc_exchange/
  Type: GDaemonMount
```

```
$ gio list smb://nasac-evs2.unige.ch/hpc_exchange/
backup

$ gio list smb://nasac-evs2.unige.ch/hpc_exchange/backup
toto
titi
tata.txt

$ gio cp smb://nasac-evs2.unige.ch/hpc_exchange/backup/tata /tmp

...
```

CVMFS

All the compute nodes of our clusters have CernVM-FS client installed. CernVM-FS, the CernVM File System (also known as CVMFS), is a file distribution service that is particularly well suited to distribute software installations across a large number of systems world-wide in an efficient way.

A couple of repository are mounted on the compute and login node such as:

- atlas.cern.ch
- atlas-condb.cern.ch
- atlas-nightlies.cern.ch
- sft.cern.ch
- grid.cern.ch

The content is mounted using autofs under the path `/cvmfs`. It means that the root directory `/cvmfs` may appears empty as long as you didn't access explicitly one of the child directory. Doing so will mount the repository for a couple of minutes and unmount it automatically.

Other flaghship repository available without further configuration:

- unpacked.cern.ch
- singularity.opensciencegrid.org (container registry)
- software.eessi.io (

```
[root@node002 ~]# ls /cvmfs
[root@node002 ~]# ls -dl /cvmfs/grid.cern.ch
drwxrwxr-x 56 cvmfs cvmfs 2 Feb  1  2010 /cvmfs/grid.cern.ch
[root@node002 ~]# ls /cvmfs/
cvmfs-config.cern.ch  grid.cern.ch
```

The EESSI did a nice tutorial about CVMFS readable on multixscale git repo.

Robinhood

Robinhood Policy Engine is a versatile tool to manage contents of large file systems. It daily scans the

scratch beegfs filesystems. It makes it possible to schedule mass action on filesystem entries by defining attribute-based policies.

We are working on the newer functionality needed to enforce our scratch data retention policy, the report are out of date until further notice

Policies

Each file matchinga policy will be recorded in a robinhood report and accessible by the owner.

LargeDir: Triggered if the directory contains more than 500 files

The more files a directory contains, the more impact it has on the performance of the filesystem and therefore on other users. So we stronghly encourage you to avoid The LargeDir .

OldFiles: Triggered if a file has not been accessed for more than 60 days

The colder the data, the less likely it is to be reused. This rule is widely verified on most storage systems. As scratch is not a permanent storage, cold data tends to take up space unnecessarily. Some files have not been opened/modified since 2013. What is the probability that it will be used again soon. (close to 0%)

Eventually, we may apply certain actions to policies, such as deleting old and unused files, to free up space and allow users who need it to continue their projects. So stay tuned to the HPC team's communication, to the best practices described in our documentation and to the evolution of robinhood.

Report

If some files have been triggered by robinhood, a report will be displayed on the connection node:

```
(baobab)-[terminator@login2~]$

################################################################################
                                Robinhood Report
################################################################################

Cluster storage usage:

       USER >     /home     | /srv/beegfs/scratch
       root >  107.48 GiB   |    1.79 TiB
```

```
Policies report:
    * LargeDir: 173
      Triggered if the directory contains more than 500 files

    * OldFiles: 12307
      Triggered if a file has not been accessed for more than 60 days



Please check /srv/beegfs/scratch/log/robinhood/report/t/terminator.log
for more details and clean them up if  possible.

##############################################################################
###
```

This is not a decorative item and should be taken seriously, this report is intended to inform you of files that do not comply with the storage policy. This list of files should get your attention and prompt you to sort through your files, delete or migrate your data to permanent storage, or reduce the number of files in a directory to avoid a drop in performance.

1) https://hpc-community.unige.ch/t/some-directory-permission-seem-to-change-on-their-own/1050/2
2) https://hpc-community.unige.ch/t/storage-on-baobab/781
3) How to access external storage from Baobab

From:
https://doc.eresearch.unige.ch/ - **eResearch Doc**

Permanent link:
https://doc.eresearch.unige.ch/hpc/storage_on_hpc?rev=1744020158

Last update: **2025/06/11 12:27**