

# Storage

## Introduction

There are two main shared file systems on Salomon cluster, the [HOME](storage.html#home) and SCRATCH. All login and compute nodes may access same data on shared filesystems. Compute nodes are also equipped with local (non-shared) scratch, ramdisk and tmp filesystems.

## Policy (in a nutshell)

Use HOME for your most valuable data and programs. Use WORK for your large project files Use TEMP for large scratch data.

Do not use for archiving!

## Archiving

Please don't use shared filesystems as a backup for large amount of data or long-term archiving mean. The academic staff and students of research institutions in the Czech Republic can use CESNET storage service, which is available via SSHFS.

## Shared Filesystems

Salomon computer provides two main shared filesystems, the HOME filesystem and the SCRATCH filesystem. The SCRATCH filesystem is partitioned to WORK and TEMP workspaces. The HOME filesystem is realized as a tiered NFS disk storage. The SCRATCH filesystem is realized as a parallel Lustre filesystem. Both shared file systems are accessible via the Infiniband network. Extended ACLs are provided on both HOME/SCRATCH filesystems for the purpose of sharing data with other users using fine-grained control.

### HOME filesystem

The HOME filesystem is realized as a Tiered filesystem, exported via NFS. The first tier has capacity 100TB, second tier has capacity 400TB. The filesystem is available on all login and computational nodes. The Home filesystem hosts the HOME workspace.

## SCRATCH filesystem

The architecture of Lustre on Salomon is composed of two metadata servers (MDS) and six data/object storage servers (OSS). Accessible capacity is 1.69 PB, shared among all users. The SCRATCH filesystem hosts the WORK and TEMP workspaces.

Configuration of the SCRATCH Lustre storage

- SCRATCH Lustre object storage
  - Disk array SFA12KX
  - 540 4TB SAS 7.2krpm disks
  - 54 OSTs of 10 disks in RAID6 (8+2)
  - 15 hot-spare disks
  - 4x 400GB SSD cache
- SCRATCH Lustre metadata storage
  - Disk array EF3015
  - 12 600GB SAS 15krpm disks

## Understanding the Lustre Filesystems

(source <http://www.nas.nasa.gov>)

A user file on the Lustre filesystem can be divided into multiple chunks (stripes) and stored across a subset of the object storage targets (OSTs) (disks). The stripes are distributed among the OSTs in a round-robin fashion to ensure load balancing.

When a client (a compute node from your job) needs to create or access a file, the client queries the metadata server (MDS) and the metadata target (MDT) for the layout and location of the file's stripes. Once the file is opened and the client obtains the striping information, the MDS is no longer involved in the file I/O process. The client interacts directly with the object storage servers (OSSes) and OSTs to perform I/O operations such as locking, disk allocation, storage, and retrieval.

If multiple clients try to read and write the same part of a file at the same time, the Lustre distributed lock manager enforces coherency so that all clients see consistent results.

There is default stripe configuration for Salomon Lustre filesystems. However, users can set the following stripe parameters for their own directories or files to get optimum I/O performance:

1. `stripe_size`: the size of the chunk in bytes; specify with k, m, or g to use units of KB, MB, or GB, respectively; the size must be an even multiple of 65,536 bytes; default is 1MB for all Salomon Lustre filesystems

2. `stripe_count` the number of OSTs to stripe across; default is 1 for Salomon Lustre filesystems one can specify -1 to use all OSTs in the filesystem.
3. `stripe_offset` The index of the OST where the first stripe is to be placed; default is -1 which results in random selection; using a non-default value is NOT recommended.

Setting stripe size and stripe count correctly for your needs may significantly impact the I/O performance you experience.

Use the `lfs getstripe` for getting the stripe parameters. Use the `lfs setstripe` command for setting the stripe parameters to get optimal I/O performance The correct stripe setting depends on your needs and file access patterns.

```
$ lfs getstripe dir|filename
$ lfs setstripe -s stripe_size -c stripe_count -o stripe_offset dir|filename
```

Example:

```
$ lfs getstripe /scratch/work/user/username
/scratch/work/user/username
stripe_count: 1 stripe_size: 1048576 stripe_offset: -1
```

```
$ lfs setstripe -c -1 /scratch/work/user/username/
$ lfs getstripe /scratch/work/user/username/
/scratch/work/user/username/
stripe_count: -1 stripe_size: 1048576 stripe_offset: -1
```

In this example, we view current stripe setting of the `/scratch/username/` directory. The stripe count is changed to all OSTs, and verified. All files written to this directory will be striped over all (54) OSTs

Use `lfs check OSTs` to see the number and status of active OSTs for each filesystem on Salomon. Learn more by reading the man page

```
$ lfs check osts
$ man lfs
```

## Hints on Lustre Stripping

Increase the `stripe_count` for parallel I/O to the same file.

When multiple processes are writing blocks of data to the same file in parallel, the I/O performance for large files will improve when the `stripe_count` is set to a larger value. The stripe count sets the number of OSTs the file will be written to. By default, the stripe count is set to 1. While this default setting provides for efficient access of metadata (for example to support the `ls -l` command), large files should use stripe counts of greater than 1. This will increase the aggregate I/O bandwidth by using multiple OSTs in parallel instead of just one. A rule of

thumb is to use a stripe count approximately equal to the number of gigabytes in the file.

Another good practice is to make the stripe count be an integral factor of the number of processes performing the write in parallel, so that you achieve load balance among the OSTs. For example, set the stripe count to 16 instead of 15 when you have 64 processes performing the writes.

Using a large stripe size can improve performance when accessing very large files

Large stripe size allows each client to have exclusive access to its own part of a file. However, it can be counterproductive in some cases if it does not match your I/O pattern. The choice of stripe size has no effect on a single-stripe file.

Read more on [http://wiki.lustre.org/manual/LustreManual20\\_HTML/ManagingStripingFreeSpace.html](http://wiki.lustre.org/manual/LustreManual20_HTML/ManagingStripingFreeSpace.html)

## >Disk usage and quota commands

User quotas on the Lustre file systems (SCRATCH) can be checked and reviewed using following command:

```
$ lfs quota dir
```

Example for Lustre SCRATCH directory:

```
$ lfs quota /scratch
```

Disk quotas for user user001 (uid 1234):

Filesystem	kbytes	quota	limit	grace	files	quota	limit	grace	
/scratch	8	0	1000000000000		-	3	0	0	-

Disk quotas for group user001 (gid 1234):

Filesystem	kbytes	quota	limit	grace	files	quota	limit	grace
/scratch	8	0	0	-	3	0	0	-

In this example, we view current quota size limit of 100TB and 8KB currently used by user001.

HOME directory is mounted via NFS, so a different command must be used to obtain quota information:

```
$ quota
```

Example output:

```
$ quota
```

Disk quotas for user vop999 (uid 1025):

Filesystem	blocks	quota	limit	grace	files	quota	limit	grace
home-nfs-ib.salomon.it4i.cz:/home	28	0	250000000		10	0	500000	

To have a better understanding of where the space is exactly used, you can use following command to find out.

```
$ du -hs dir
```

Example for your HOME directory:

```
$ cd /home
$ du -hs * .[a-zA-z0-9]* | grep -E "[0-9]*G|[0-9]*M" | sort -hr
258M      cuda-samples
15M       .cache
13M       .mozilla
5,5M      .eclipse
2,7M      .idb_13.0_linux_intel64_app
```

This will list all directories which are having MegaBytes or GigaBytes of consumed space in your actual (in this example HOME) directory. List is sorted in descending order from largest to smallest files/directories.

To have a better understanding of previous commands, you can read manpages.

```
$ man lfs
```

```
$ man du
```

## Extended Access Control List (ACL)

Extended ACLs provide another security mechanism beside the standard POSIX ACLs which are defined by three entries (for owner/group/others). Extended ACLs have more than the three basic entries. In addition, they also contain a mask entry and may contain any number of named user and named group entries.

ACLs on a Lustre file system work exactly like ACLs on any Linux file system. They are manipulated with the standard tools in the standard manner. Below, we create a directory and allow a specific user access.

```
[vop999@login1.salomon ~]$ umask 027
[vop999@login1.salomon ~]$ mkdir test
[vop999@login1.salomon ~]$ ls -ld test
drwxr-x--- 2 vop999 vop999 4096 Nov  5 14:17 test
[vop999@login1.salomon ~]$ getfacl test
# file: test
# owner: vop999
# group: vop999
user::rwx
group::r-x
other::---
```

```
[vop999@login1.salomon ~]$ setfacl -m user:johnsm:rwx test
[vop999@login1.salomon ~]$ ls -ld test
drwxrwx---+ 2 vop999 vop999 4096 Nov  5 14:17 test
[vop999@login1.salomon ~]$ getfacl test
# file: test
# owner: vop999
# group: vop999
user::rwx
user:johnsm:rwx
group::r-x
mask::rwx
other::---
```

Default ACL mechanism can be used to replace setuid/setgid permissions on directories. Setting a default ACL on a directory (-d flag to setfacl) will cause the ACL permissions to be inherited by any newly created file or subdirectory within the directory. Refer to this page for more information on Linux ACL:

[http://www.vanemery.com/Linux/ACL/POSIX\\_ACL\\_on\\_Linux.html](http://www.vanemery.com/Linux/ACL/POSIX_ACL_on_Linux.html)

## Shared Workspaces

### HOME

Users home directories /home/username reside on HOME filesystem. Accessible capacity is 0.5PB, shared among all users. Individual users are restricted by filesystem usage quotas, set to 250GB per user. >If 250GB should prove as insufficient for particular user, please contact support, the quota may be lifted upon request.

The HOME filesystem is intended for preparation, evaluation, processing and storage of data generated by active Projects.

The HOME should not be used to archive data of past Projects or other unrelated data.

The files on HOME will not be deleted until end of the users lifecycle.

The workspace is backed up, such that it can be restored in case of catastrophic failure resulting in significant data loss. This backup however is not intended to restore old versions of user data or to restore (accidentaly) deleted files.

HOME workspace Accesspoint /home/username Capacity 0.5PB Throughput 6GB/s User quota 250GB Protocol NFS, 2-Tier ### WORK

The WORK workspace resides on SCRATCH filesystem. Users may create subdirectories and files in directories /**scratch/work/user/username** and /**scratch/work/project/projectid**. The /scratch/work/user/username is

private to user, much like the home directory. The `/scratch/work/project/projectid` is accessible to all users involved in project `projectid`. >

The WORK workspace is intended to store users project data as well as for high performance access to input and output files. All project data should be removed once the project is finished. The data on the WORK workspace are not backed up.

Files on the WORK filesystem are **persistent** (not automatically deleted) throughout duration of the project.

The WORK workspace is hosted on SCRATCH filesystem. The SCRATCH is realized as Lustre parallel filesystem and is available from all login and computational nodes. Default stripe size is 1MB, stripe count is 1. There are 54 OSTs dedicated for the SCRATCH filesystem.

Setting stripe size and stripe count correctly for your needs may significantly impact the I/O performance you experience.

WORK workspace Accesspoints `/scratch/work/user/username /scratch/work/user/projectid`  
Capacity 1.6P Throughput 30GB/s User quota 100TB Default stripe size 1MB  
Default stripe count 1 Number of OSTs 54 Protocol Lustre ### TEMP

The TEMP workspace resides on SCRATCH filesystem. The TEMP workspace accesspoint is `/scratch/temp`. Users may freely create subdirectories and files on the workspace. Accessible capacity is 1.6P, shared among all users on TEMP and WORK. Individual users are restricted by filesystem usage quotas, set to 100TB per user. The purpose of this quota is to prevent runaway programs from filling the entire filesystem and deny service to other users. >If 100TB should prove as insufficient for particular user, please contact support, the quota may be lifted upon request.

The TEMP workspace is intended for temporary scratch data generated during the calculation as well as for high performance access to input and output files. All I/O intensive jobs must use the TEMP workspace as their working directory.

Users are advised to save the necessary data from the TEMP workspace to HOME or WORK after the calculations and clean up the scratch files.

Files on the TEMP filesystem that are **not accessed for more than 90 days** will be automatically **deleted**.

The TEMP workspace is hosted on SCRATCH filesystem. The SCRATCH is realized as Lustre parallel filesystem and is available from all login and computational nodes. Default stripe size is 1MB, stripe count is 1. There are 54 OSTs dedicated for the SCRATCH filesystem.

Setting stripe size and stripe count correctly for your needs may significantly impact the I/O performance you experience.

TEMP workspace Accesspoint `/scratch/temp` Capacity 1.6P Throughput 30GB/s User quota 100TB Default stripe size 1MB Default stripe count 1

Number of OSTs 54 Protocol Lustre

## RAM disk

Every computational node is equipped with filesystem realized in memory, so called RAM disk.

Use RAM disk in case you need really fast access to your data of limited size during your calculation. Be very careful, use of RAM disk filesystem is at the expense of operational memory.

The local RAM disk is mounted as /ramdisk and is accessible to user at /ramdisk/\$PBS\_JOBID directory.

The local RAM disk filesystem is intended for temporary scratch data generated during the calculation as well as for high performance access to input and output files. Size of RAM disk filesystem is limited. Be very careful, use of RAM disk filesystem is at the expense of operational memory. It is not recommended to allocate large amount of memory and use large amount of data in RAM disk filesystem at the same time.

The local RAM disk directory /ramdisk/\$PBS\_JOBID will be deleted immediately after the calculation end. Users should take care to save the output data from within the jobscript.

RAM disk Mountpoint /ramdisk Accesspoint /ramdisk/\$PBS\_JOBID Capacity 120 GB Throughput over 1.5 GB/s write, over 5 GB/s read, single thread over 10 GB/s write, over 50 GB/s read, 16 threads

User quota none

Summary

---

Mountpoint	Usage	Protocol	Capacity	Throughput	Limitations	Access	Services
/home home directory	NFS, 2-Tier 0.5 PB 6	GB/s Qu	ota 250GB Com	pute and logi	n nodes back	ed up	
/scratch/work large project files	Lustre 1.69 PB 3	0 GB/s Qu	ota Comp lo-gin	ute and none 1TB	nodes		



Mountpoint	Usage	Protocol	Net Ca- pac- ity	Through- put	Limitations	Access	Services
/scratch/temp job temporary data	Lustre 1.69 PB 3	0 GB/s Qu	ota 100TB Com	pute and logi	n nodes file	s older 90 days removed	
/ramdisk job temporary data, node local	local 120GB 9	0 GB/s no	ne Com	pute nodes	purg ed after job ends		