

# Data storage, transfer and sharing

[damien.francois@uclouvain.be](mailto:damien.francois@uclouvain.be)

november 2024

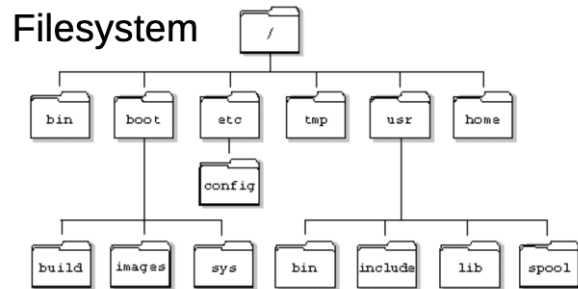
## Goal of this session:

Share tools, tips and tricks related to the **storage, transfer,** and **sharing** of scientific data on the clusters.

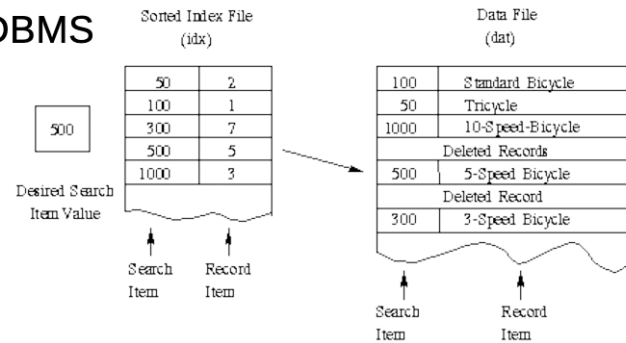
# Data storage

- File
  - Filesystems
  - File formats
  - Common problems with files
- Object storage
- Database systems

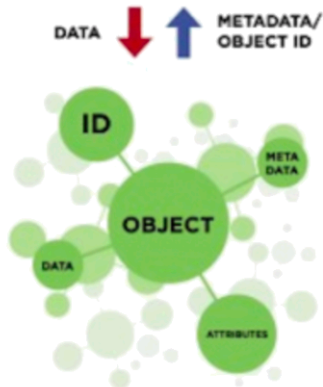
# Data storage paradigms



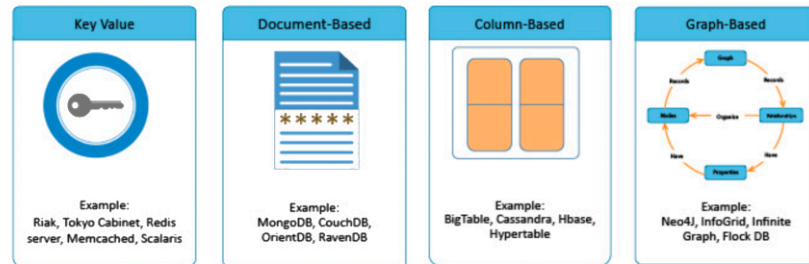
## RDBMS



## Objects store



## NoSQL





# Filesystems

Technology (method and data structure) used by the operating system to store and retrieve files.

Can be

- *local* on disk or in RAM, viewed only from one server, or
- *shared* through the network, visible from multiple servers.

# Shared filesystems

## Network filesystems

- one server multiple clients (NFS, CIFS)
- typically used for the `$HOME` directories

## Parallel filesystems

- multiple servers multiple clients (BeeGFS, GPFS, Lustre)
- typically used for the global scratch `$GLOBALSCRATCH`

# Lemaitre4

## RACK 1

- Switch Infiniband
- Switch Ethernet
- Storage Home
  
- Storage Data Scratch
- Server Login
- Server Front
  
- Chassis Bi-Twin : 4 Nodes
- Chassis Bi-Twin : 4 Nodes
- Chassis Bi-Twin : 4 Nodes
- Chassis Bi-Twin : 4 Nodes
- Chassis Bi-Twin : 4 Nodes



## RACK 2

- Switch Ethernet
- Storage MetaData Scratch
  
- Storage Data Scratch
- Server Login
- Server Front
  
- Chassis Bi-Twin : 4 Nodes
- Chassis Bi-Twin : 4 Nodes
- Chassis Bi-Twin : 4 Nodes
- Chassis Bi-Twin : 4 Nodes
- Chassis Bi-Twin : 4 Nodes



# Lemaitre4 filesystems

```
[dfr@lm4-f001 ~]$ df -khT -x tmpfs
```

Filesystem	Type	Size	Used	Avail	Use%	Mounted on
/dev/sda3	xfs	48G	6.3G	42G	14%	/
/dev/sda2	xfs	1006M	202M	805M	21%	/boot
/dev/sda1	vfat	599M	5.8M	594M	1%	/boot/efi
/dev/sda4	xfs	16G	6.9G	9.2G	44%	/tmp
/dev/sda6	xfs	378G	2.7G	375G	1%	/localscratch
gw-ucl:/CECI/gateway/proj	nfs	32T	24T	8.1T	75%	/CECI/proj
lm4-n001-ib:/soft/localsoft/RedHat-8_25-17-1_Infiniband	nfs	1.3T	664G	617G	52%	/opt/sw/arch
beegfs_nodev	beegfs	318T	267T	51T	84%	/globalscratch
10.44.3.1:/home	nfs4	22T	3.3T	19T	16%	/home

## Source:

- `/dev/sd...` → local disk
- `<machine>:<path>` → NFS
- other (e.g. `beegfs_nodev`) → specific filesystem

# Lemaitre4 filesystems

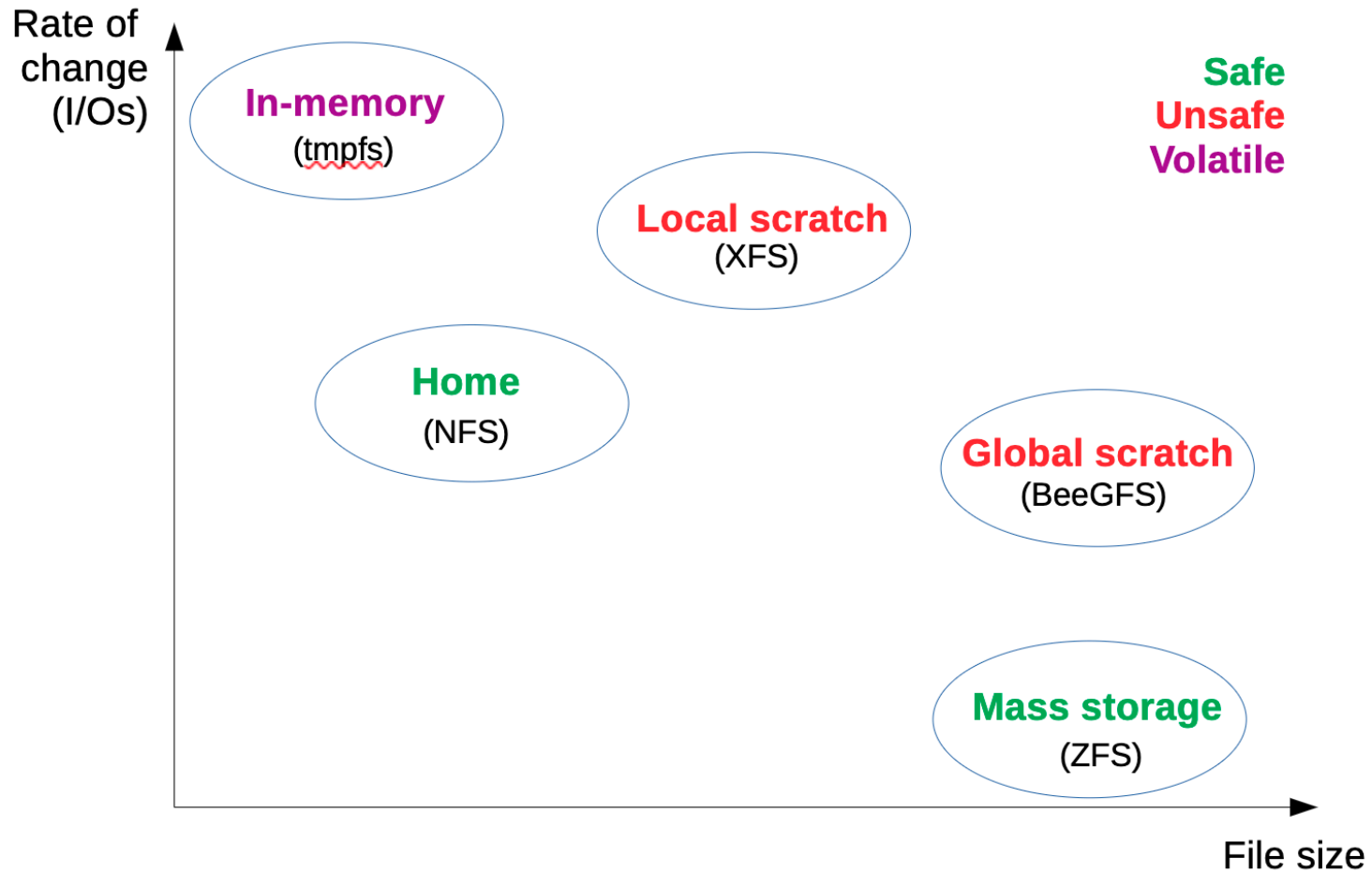
```
[dfr@lm4-f001 ~]$ df -khT -x tmpfs -i
```

Filesystem	Type	Inodes	IUsed	IFree	IUse%	Mounted on
/dev/sda3	xfs	24M	208K	24M	1%	/
/dev/sda2	xfs	512K	19	512K	1%	/boot
/dev/sda1	vfat	0	0	0	-	/boot/efi
/dev/sda4	xfs	8.0M	20K	8.0M	1%	/tmp
/dev/sda6	xfs	189M	12	189M	1%	/localscratch
gw-ucl:/CECI/gateway/proj	nfs	108M	39M	69M	37%	/CECI/proj
lm4-n001-ib:/soft/localsoft/RedHat-8_25-17-1_Infiniband	nfs	128M	3.3M	125M	3%	/opt/sw/arch
beegfs_nodev	beegfs	0	0	0	-	/globalscratch
10.44.3.1:/home	nfs4	37G	28M	37G	1%	/home

## INodes:

- entries in the file datastructure
- roughly proportional to the number of files (and their size)
- **often disregarded but more important than volume!**

# What filesystem for what usage



# File formats

Standard way information is organized and encoded in a file

Can be

- *text*, readable by a human, but space-inefficient
- *binary*, readable by dedicated software, often compressed.





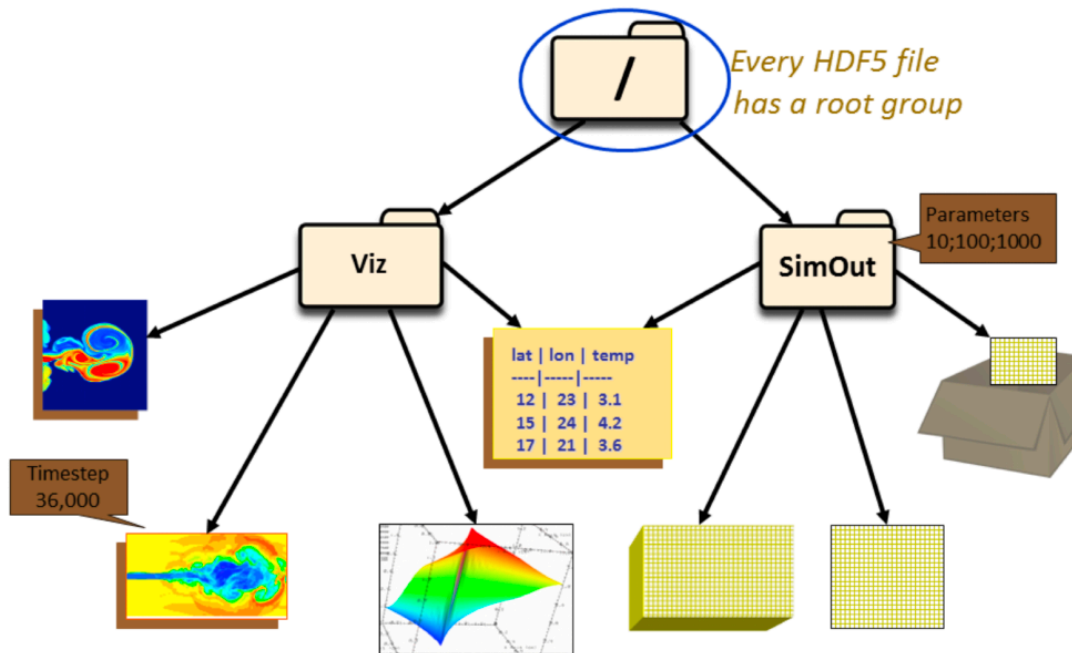
# Text file formats

```
processes.txt - Notepad
File Edit Format View Help
__NounName Name Handles VM WS PM NPM Path Company CPU FileVersion ProductVer
"Process" "AcDeskBandHpr" "127" "85278720" "10047488" "4362240" "14224" "C:\Program Files (x86)\Lei
"Process" "AcPrfMgrSvc" "167" "72732672" "11051008" "4956160" "17128" "" "" "" "" "" "" "" "" "" "" "" "" ""
"Process" "AcSvc" "421" "127533056" "17551360" "10117120" "26960" "" "" "" "" "" "" "" "" "" "" "" "" ""
"Process" "armsvc" "84" "45129728" "4509696" "1601536" "8528" "" "" "" "" "" "" "" "" "" "" "" "" ""
"Process" "btwdins" "127" "60628992" "7720960" "3743744" "10096" "" "" "" "" "" "" "" "" "" "" "" "" ""
"Process" "CamMute" "111" "40837120" "5312512" "1789952" "9440" "" "" "" "" "" "" "" "" "" "" "" "" ""
"Process" "CcmExec" "1433" "329342976" "89608192" "58417152" "58432" "" "" "" "" "" "" "" "" "" "" "" "" ""
"Process" "conhost" "35" "25186304" "4136960" "2048000" "5168" "" "" "" "" "" "" "" "" "" "" "" "" ""
"Process" "conhost" "45" "52461568" "6311936" "3215360" "6360" "C:\windows\system32\conhost.exe" "M
"Process" "csrss" "1145" "55078912" "5623808" "2977792" "17768" "" "" "" "" "" "" "" "" "" "" "" "" ""
"Process" "csrss" "628" "96055296" "17383424" "20799488" "28704" "" "" "" "" "" "" "" "" "" "" "" "" ""
"Process" "CxAudMsg64" "98" "57937920" "6680576" "7385088" "8832" "" "" "" "" "" "" "" "" "" "" "" "" ""
"Process" "daemonu" "413" "73506816" "8957952" "5935104" "20132" "" "" "" "" "" "" "" "" "" "" "" "" ""
"Process" "DcaSvc" "642" "64344064" "13996032" "10178560" "27904" "" "" "" "" "" "" "" "" "" "" "" "" ""
"Process" "DcaTray" "644" "220614656" "35368960" "33832960" "34628" "C:\Program Files (x86)\DirectA
"Process" "dwm" "108" "84504576" "10313728" "5709824" "10024" "C:\windows\system32\Dwm.exe" "Microso
"Process" "EvtEng" "284" "105017344" "20992000" "12906496" "22248" "" "" "" "" "" "" "" "" "" "" "" "" ""
"Process" "EXCEL" "384" "316968960" "42233856" "27746304" "48584" "C:\Program Files (x86)\Microsoft
"Process" "explorer" "1153" "352845824" "75698176" "52506624" "94040" "C:\windows\Explorer.EXE" "Mi
"Process" "fmapp" "30" "53719040" "6168576" "3743744" "6960" "C:\Program Files\CONEXANT\ForteConfig
"Process" "FwcAgent" "555" "59432960" "12111872" "9289728" "22248" "" "" "" "" "" "" "" "" "" "" "" "" ""
"Process" "hkcmd" "84" "71577600" "8040448" "3964928" "8048" "C:\windows\system32\hkcmd.exe" "Intel
"Process" "ibmpmsvc" "61" "44564480" "4259840" "2301952" "6568" "" "" "" "" "" "" "" "" "" "" "" "" ""
"Process" "Idle" "0" "24576" "0" "0" "" "" "" "" "" "" "" "" "" "" "" "" ""
"Process" "iexplore" "774" "304881664" "79278080" "77529088" "82152" "C:\Program Files (x86)\Intern
"Process" "iexplore" "639" "157421568" "22876160" "12095488" "38648" "C:\Program Files (x86)\Intern
"Process" "igfxpers" "203" "83292160" "10944512" "5844992" "10808" "C:\windows\System32\igfxpers.ex
"Process" "lms" "112" "41308160" "5484544" "2117632" "9800" "" "" "" "" "" "" "" "" "" "" "" "" ""
"Process" "lsass" "1545" "75075584" "27377664" "18477056" "47272" "" "" "" "" "" "" "" "" "" "" "" "" ""
"Process" "lsm" "213" "19779584" "5664768" "3756032" "7600" "" "" "" "" "" "" "" "" "" "" "" "" ""
"Process" "micmute" "110" "48328704" "5672960" "6172672" "11120" "" "" "" "" "" "" "" "" "" "" "" "" ""
"Process" "mscorsvw" "105" "73936896" "14417920" "8912896" "11184" "" "" "" "" "" "" "" "" "" "" "" "" ""
"Process" "MsitBLSHA" "118" "45617152" "8912896" "4481024" "10328" "" "" "" "" "" "" "" "" "" "" "" "" ""
"Process" "msitcertsvc" "712" "565702656" "33861632" "40435712" "41284" "" "" "" "" "" "" "" "" "" "" "" "" ""
"Process" "MsitTpmSvc" "84" "43319296" "10129408" "6975488" "7568" "" "" "" "" "" "" "" "" "" "" "" "" ""
"Process" "MsMpEng" "503" "238940160" "87953408" "117288960" "47112" "" "" "" "" "" "" "" "" "" "" "" "" ""
"Process" "MSOIDSVC" "612" "106561536" "21938176" "14983168" "27648" "" "" "" "" "" "" "" "" "" "" "" "" ""
"Process" "MSOIDSVC" "72" "36134912" "4841472" "2621440" "6080" "" "" "" "" "" "" "" "" "" "" "" "" ""
"Process" "msseces" "341" "156336128" "23216128" "10706944" "25232" "C:\Program Files\Microsoft Seci
"Process" "NisSrv" "261" "80003072" "4771840" "9138176" "18000" "" "" "" "" "" "" "" "" "" "" "" "" ""
"Process" "nusb3mon" "89" "75804672" "5914624" "2224128" "10328" "C:\Program Files (x86)\Renesas El...
```

# Binary file formats

September 23, 2016

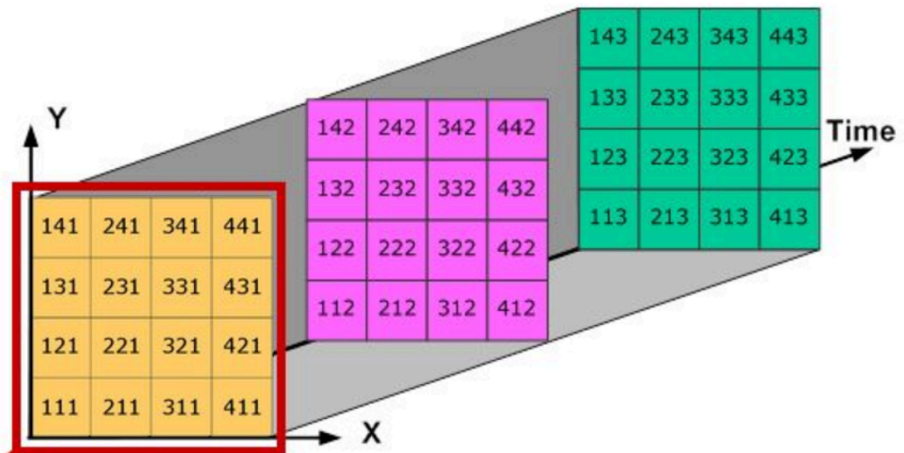
Introduction to HDF5



There are two groups in the HDF5 file depicted above: Viz and SimOut. Under the Viz group are a variety of images and a table that is shared with the SimOut group. The SimOut group contains a 3-dimensional array, a 2-dimensional array and a link to a 2-dimensional array in another HDF5 file.

# Binary file formats

```
netcdf mynetcdf {  
  dimensions:  
    x=4;  
    y=4;  
    time=UNLIMITED;  
  variables:  
    float x(x);  
    float y(y);  
    int time(time);  
    float temperature(time,x,y);  
  data:  
    x = 10, 20, 30, 40;  
    y = 110, 120, 130, 140;  
    time = 31, 59, 90;  
  Temperature = 111, 211, 311, 411, 121, 221, 321,  
    421, 131, 231, 331, 431, 141, 241, 341, 441;  
}
```



time = 1

x = 1 to 4

y = 1 to 4

# What file format for what usage

## Meta data:

- Configuration file: INI, YAML
- Result with context information: JSON

## Data:

- Small data (kB): CSV
- Medium data (MB): compressed CSV
- Large data (GB): netCDF, HDF5, ASDF, Zarr
- Huge data (TB): Object store
- Huge number of small files ( $10^6$ \*kB): Database

# The problems with files

Filesystems are not designed

- to host millions of files
- to manage concurrent accesses diligently
- to finely organise sharing of the files

# Problem 1: ZOT files

"Zillions of tiny" files

- Over-consume resources
  - Inode spaces is finite on most filesystems
  - Minimal chunk size often large on HPC filesystems
- Makes system slower (`ls`, `rsync`, `rm` etc.)
  - Inodes operations cannot be buffered/cached easily
  - Event simple `ls -l` can put heavy burden on MD servers  
slowing all the operations
- Easy to loose control

# Problem 1: ZOT files (solutions)

Write a single file:

- TAR archive
- Singularity container

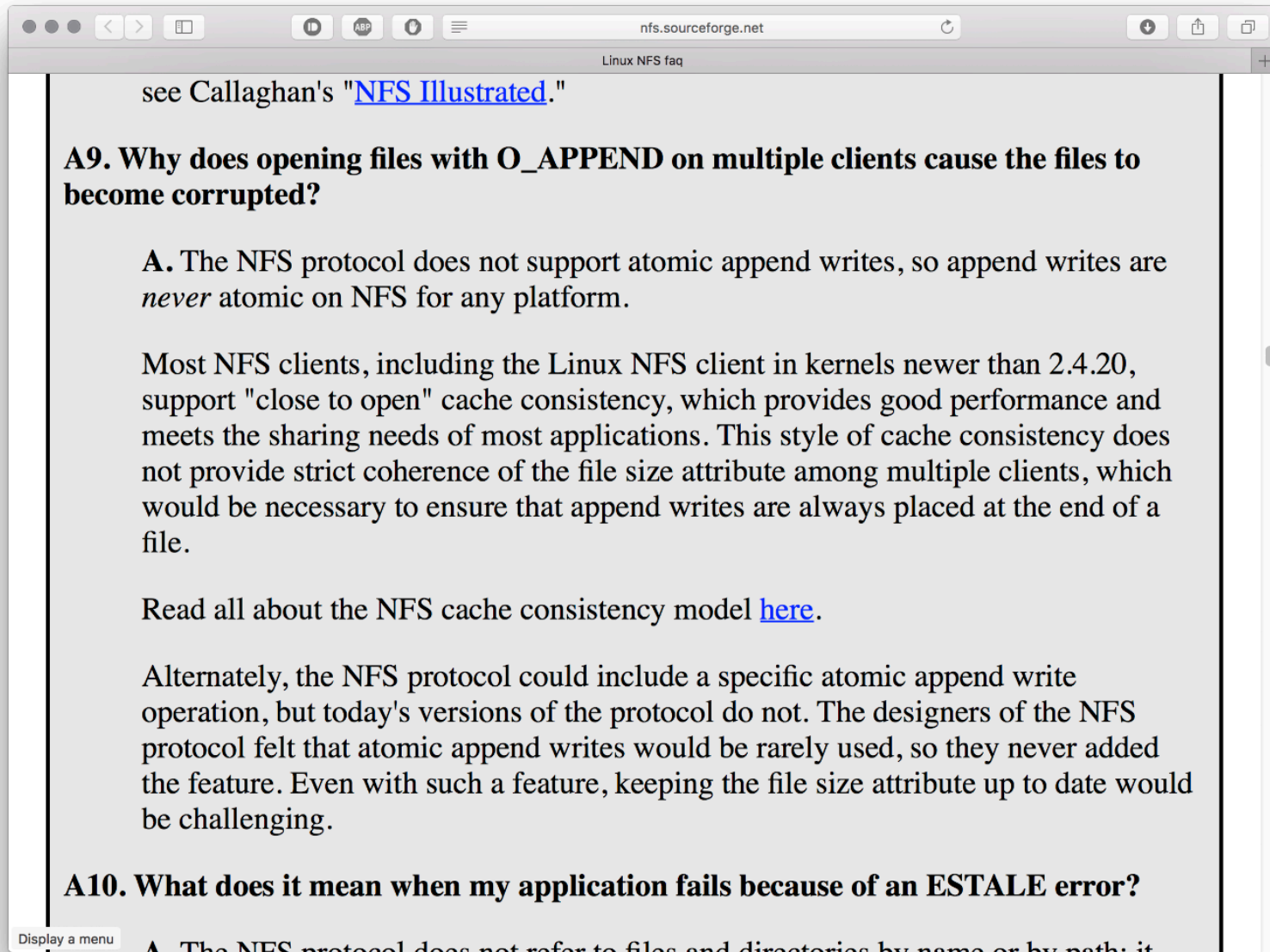
Write them to local filesystems

- Even better to memory filesystems

Even better to another storage type:

- Object store (rich efficient meta data)
- Database (strong structure)

# Problem 2: Concurrent access



see Callaghan's "[NFS Illustrated](#)."

**A9. Why does opening files with O\_APPEND on multiple clients cause the files to become corrupted?**

**A.** The NFS protocol does not support atomic append writes, so append writes are *never* atomic on NFS for any platform.

Most NFS clients, including the Linux NFS client in kernels newer than 2.4.20, support "close to open" cache consistency, which provides good performance and meets the sharing needs of most applications. This style of cache consistency does not provide strict coherence of the file size attribute among multiple clients, which would be necessary to ensure that append writes are always placed at the end of a file.

Read all about the NFS cache consistency model [here](#).

Alternately, the NFS protocol could include a specific atomic append write operation, but today's versions of the protocol do not. The designers of the NFS protocol felt that atomic append writes would be rarely used, so they never added the feature. Even with such a feature, keeping the file size attribute up to date would be challenging.

**A10. What does it mean when my application fails because of an ESTALE error?**

**A.** The NFS protocol does not refer to files and directories by name or by path; it



## **Problem 2: Concurrent access (solutions)**

- Use a library for (organised) parallel writes (e.g. netCDF)
- Write to local filesystems and merge afterwards
- Use a database

## Problem 3: sharing

- Everyone must have access to the same computer
- UNIX permissions are not so flexible
  - Groups must be set by admins
  - No organization of groups

## Problem 3: sharing (solutions)

- ~~Dumb it down: chmod 777~~
- Abuse UNIX permissions
- Use an Object store

# Object store

data storage technology that manages data as objects that include the data itself, a variable amount of metadata, and a globally unique identifier



Useful for web applications but coming to scientific world  
Access with REST API (through HTTP)

# Object store on Lumi

The screenshot displays the LUMI Object storage - LUMI-O documentation page. The browser address bar shows the URL `docs.lumi-supercomputer.eu/hardware/storage/lumi/`. The page title is "Object storage - LUMI-O". A search bar is located in the top right corner.

**Hardware**

- Overview
- Interconnect

**Compute nodes**

- CPU nodes - LUMI-C
- GPU nodes - LUMI-G
- Data analytics nodes - LUMI-D
- GPU Early Access Platform

**Storage**

- Main storage - LUMI-P
- Flash storage - LUMI-F
- Object storage - LUMI-O**

**Auxiliary platforms**

- Container Orchestration Platform - LUMI-K

The main content area contains the following text:

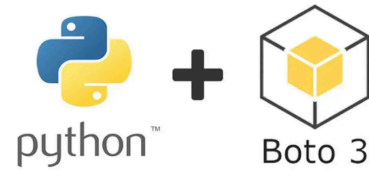
The LUMI-O object store offers a total of 30 PB storage for storing, sharing, and staging of data.

In an object-based storage, data is managed as objects instead of being organized as files in a directory hierarchy.

Within your object storage project space you can create buckets. These buckets will store objects with metadata associated to these objects.

The diagram illustrates a "Project" space containing three "Buckets" (Bucket 1, Bucket 2, Bucket 3). Each bucket contains objects represented by colored cubes (red, green, purple). An arrow points to one of the objects with the label "object".

## Access with code



## Access with command line



## Setup your own



# Object store

## When to use

- For data that is written once and then read multiple times from multiple remote locations as a whole
- Input data for in-memory decompression
- Output files for egress or sharing

## When not to use

- When direct access to portions of a file are needed
- When data is not meant to be read sequentially

# Database systems

system that interfaces users, applications, and the database itself to capture and analyze the data



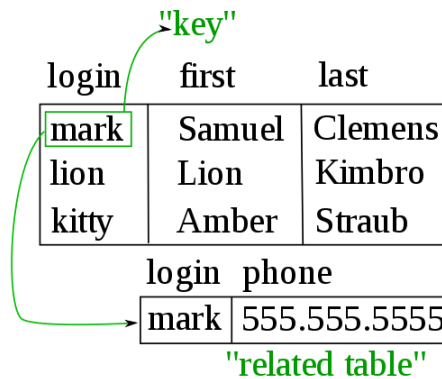
useful for enterprise data but coming to the HPC world  
access via query language and software libraries



# Relational database

## Query language : SQL

```
create table Users (login varchar(255), first varchar(255), last varchar(255));  
insert into Users values ("mark", 'Samuel", "Clemens");  
select first,last from Users where login='lion';  
select login, phone from Users join PhoneNb on Users.login=Phone.login;
```



## Setup your own



File-based database system, easy way to replace a large collection of small files with a single file.



rqlite

The lightweight, distributed relational database

No-setup server on top of SQLite that can cope with concurrent accesses.

# NoSQL

## key-value

Amazon  
DynamoDB (Beta)

ORACLE  
BERKELEY DB **11g**

 redis

---

## graph

 **Neo4j**  
the graph database

 InfiniteGraph

 sones

---

## column

 **HBASE**

 **riak**

 **Cassandra**

---

## document

 **CouchDB**  
relax

 **mongoDB**

 **elastic**

# Setup your own

## TinyDB

File-based document-oriented database system, easy way to replace a large collection of small files with a single file.

## redis

No-setup key-value database server that can cope with concurrent accesses.

## ZincSearch

No-setup document-oriented database server that can cope with concurrent accesses.

# Database

## When to use

- when you have a large collection of small files
- when you perform a lot of direct writes in a large file
- when you want to keep structure/relations between data

Many small results

## When not to use

- only sequential access
- simple matrices/vectors, etc.
- direct access on fixed-size records and no structure

# Redis example

```
redis-server.sh:  
#!/bin/bash  
#SBATCH -n1 --mem 4G  
module load redis  
hostname -s > $HOME/redisserver  
redis-server
```

```
work.sh:  
#!/bin/bash  
#SBATCH -t 10:00 -n 1 -c 4 --mem-per-cpu 4G  
#SBATCH --array 1-1000  
module load redis  
  
./myprog | redis-cli -h <(<$HOME/redisserver) -x SET res-$SLURM_TASK_ARRAY_ID
```

```
J=$(sbatch --parsable redis-server.sh)  
sbatch --dependency=after:$J work.sh
```

# Data transfer

- SCP
- RSYNC
- TAR
- Parallel RSYNC

# SCP

Simplest (and least efficient) way to copy a file to/from a remote server:

```
scp somefile lemaitre4:destination/directory  
scp lemaitre4:destination/directory/somefile .
```

Copy remote to remote:

```
scp lemaitre4:some/directory/somefile destination:  
scp -3 lemaitre4:some/directory/somefile destination:
```

Use `-3` if `source` cannot reach `destination` directly

Use `-r` ("recursive") for directories



# RSYNC

Efficient way to synchronise directories to/from a remote server:

```
rsync -va directory lemaitre4:some/directory  
rsync -va lemaitre4:some/directory .
```

Rsync will only transfer the parts of the files that changed.

Can be used to resume an interrupted **SCP** transfer:

```
scp somelargefile lemaitre4:destination/directory # Interrupted for some reason  
rsync --append somelargefile lemaitre4:destination/directory
```

# RSYNC

Progress monitoring: use

- `-v --progress` for large files
- `--info=progress2 --no-i-r` for many smaller files

Verify what will be transferred before transferring with

- `--dry-run`

Choose files to transfer with

- `--exclude`
- `--include`

Change group on the fly with

- `-g --groupmap=*:ceci_group`

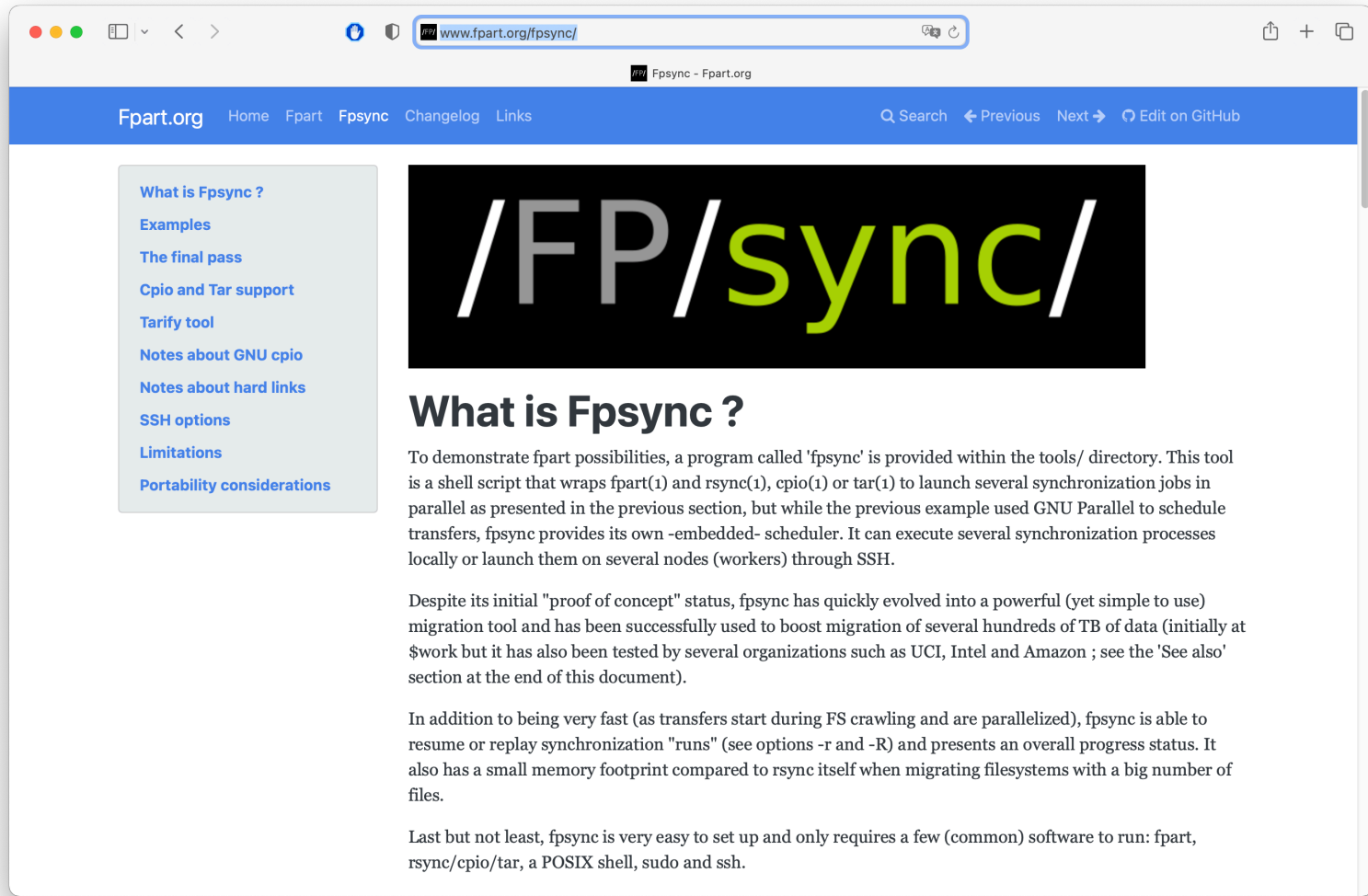
## TAR+SSH

Often, for ZOT files, creating a single large file and transferring that file will be more efficient.

```
tar czf - /path/to/data | ssh server "tar xzf - -C destination/directory"
```

This will compress and uncompress data on the fly.

# Parallel RSYNC



The screenshot shows a web browser window with the URL [www.fpart.org/fpsync/](http://www.fpart.org/fpsync/). The page has a blue header with navigation links: Home, Fpart, Fpsync, Changelog, Links, Search, Previous, Next, and Edit on GitHub. A sidebar on the left contains a list of links: What is Fpsync?, Examples, The final pass, Cpio and Tar support, Tarify tool, Notes about GNU cpio, Notes about hard links, SSH options, Limitations, and Portability considerations. The main content area features a large black banner with the text `/FP/sync/` in white and green. Below the banner is the heading 

## What is Fpsync ?

 and three paragraphs of text.

## What is Fpsync ?

To demonstrate fpart possibilities, a program called 'fpsync' is provided within the tools/ directory. This tool is a shell script that wraps fpart(1) and rsync(1), cpio(1) or tar(1) to launch several synchronization jobs in parallel as presented in the previous section, but while the previous example used GNU Parallel to schedule transfers, fpsync provides its own -embedded- scheduler. It can execute several synchronization processes locally or launch them on several nodes (workers) through SSH.

Despite its initial "proof of concept" status, fpsync has quickly evolved into a powerful (yet simple to use) migration tool and has been successfully used to boost migration of several hundreds of TB of data (initially at \$work but it has also been tested by several organizations such as UCI, Intel and Amazon ; see the 'See also' section at the end of this document).

In addition to being very fast (as transfers start during FS crawling and are parallelized), fpsync is able to resume or replay synchronization "runs" (see options -r and -R) and presents an overall progress status. It also has a small memory footprint compared to rsync itself when migrating filesystems with a big number of files.

Last but not least, fpsync is very easy to set up and only requires a few (common) software to run: fpart, rsync/cpio/tar, a POSIX shell, sudo and ssh.

```
[dfr@lm4-f001 Data]$ time scp -qr linux-6.9.8 manneback:
```

```
real    26m7.000s
user    0m3.856s
sys     0m10.972s
```

```
[dfr@lm4-f001 Data]$ time { tar czf - linux-6.9.8 | ssh manneback tar xzf - ; }
```

```
real    16m56.519s
user    0m33.286s
sys     0m4.805s
```

```
[dfr@lm4-f001 Data]$ time fpsync $HOME/Data/linux-6.9.8 manneback:$HOME/Data/linux-6.9.8
```

```
real    11m51.561s
user    0m52.537s
sys     3m16.098s
```

# Parallel RSYNC

## rclone sync

Make source and dest identical, modifying destination only.

### Synopsis

Sync the source to the destination, changing the destination only. Doesn't transfer files that are identical on source and destination, testing by size and modification time or MD5SUM. Destination is updated to match source, including deleting files if necessary (except duplicate objects, see below). If you don't want to delete files from destination, use the [copy](#) command instead.

**Important:** Since this can cause data loss, test first with the `--dry-run` or the `--interactive/-i` flag.

```
rclone sync --interactive SOURCE remote:DESTINATION
```

Note that files in the destination won't be deleted if there were any errors at any point. Duplicate objects (files with the same name, on those providers that support it) are also not yet handled.

It is always the contents of the directory that is synced, not the directory itself. So when source:path is a directory, it's the contents of source:path that are copied, not the directory name and contents. See extended explanation in the [copy](#) command if unsure.

If dest:path doesn't exist, it is created and the source:path contents go there.

### Contents




- [Synopsis](#)
- [Options](#)
- [Copy Options](#)
- [Sync Options](#)
- [Important Options](#)
- [Filter Options](#)
- [Listing Options](#)

### Gold Sponsor



Hot S3 Compatible Object Storage

### Share and Enjoy

-  [Twitter](#)
-  [Facebook](#)
-  [Reddit](#)

 Star 41,422



# Data sharing

- Personal/Sensitive data
- UNIX Permissions
- Encryption
- Nextcloud
- Dataverse



# Personal/Sensitive data

The clusters are designed for performance by default, not privacy

Responsibilities...

<b>what</b>	<b>who</b>
describing what specific protection measures to take	the project PI, or the institution's DPO
implementing protection measures	the user
making sure the infrastructure is safe and secure from external threats	the sysadmins

# Personal/Sensitive data

Four possible recommendations for personal and/or sensitive data:

- work only on local, mono-user, computer
- encrypt the data
- anonymize the data
- pseudonimize the data

# Encryption

- in *transit* -- this is always the case on clusters with SSH
- at *rest* on disks (when not processed by a job) -- it is the user's responsibility to do so, and system administrators can help set up what is needed
- at *work* in RAM (for the duration of the job) -- this is almost impossible to ensure on clusters ;

**Anonymization**

**Pseudonymization**

# Sharing with other users

JULIA Evans  
@bork

## unix permissions drawings.jvns.ca

<p>There are 3 things you can do to a file</p> <p>↓ read ↓ write ↓ execute</p>	<p>ls -l file.txt shows you permissions Here's how to interpret the output:</p> <p>rw- rw- r-- bork staff</p> <p>↑ ↑ ↑ bork (user) staff (group) ANYONE can can can read &amp; write read &amp; write read</p>	
<p>File permissions are 12 bits</p> <p>setuid sticky setgid</p> <p>000 110 110 100 user group all sticky rwx rwx rwx</p> <p>For the r/w/x bits: 1 means "allowed" 0 means "not allowed"</p>	<p>110 in binary is 6</p> <p>So rw- r-- r-- = 110 100 100 = 6 4 4</p> <p>chmod 644 file.txt means change the permissions to: rw- r-- r-- simple!</p>	<p>setuid affects executables</p> <p>\$ls -l /bin/ping rws r-x r-x root root</p> <p>↑ this means ping <u>always</u> runs as root</p> <p>setgid does 3 different unrelated things for executables, directories, and regular files</p> <p>unix! why?? it's a long story 😊 unix</p>

Note: x on a directory means traverse permission

# Sharing with other users

Make `directory` writable for the group

```
chmod g+rx directory
```

Make `file` readable by everyone

```
chmod o+r file
```

Make `directory` readable by everyone, recursively

```
chmod o+rX directory
```

# Sharing with other users

All parent directories must be *traversable*

```
[dfr@lm4-f001 Data]$ namei -l $(realpath random.dat)
f: /home/users/d/f/dfr/Data/random.dat
dr-xr-xr-x root root /
drwxr-xr-x root root home
drwxr-xr-x root root users
drwxr-xr-x root root d
drwxr-xr-x root root f
drwxr-x--x dfr dfr dfr
drwxrwx--- dfr dfr Data
-rw-rw-r-- dfr dfr random.dat
```

# Sharing with a group

See which groups you are part of:

```
[dfr@lm4-f001 ~]$ id  
uid=3000003(dfr) gid=3000003(dfr) groups=3000003(dfr),4999998(adminucl),4999999(sysadmin)
```

Change group ownership (as a regular user):

```
[dfr@lm4-f001 ~]$ ls -ld Data  
drwxrwx--- 4 dfr dfr 7 Sep 17 11:35 Data  
[dfr@lm4-f001 ~]$ chgrp adminucl Data/  
[dfr@lm4-f001 ~]$ ls -ld Data  
drwxrwx--- 4 dfr adminucl 7 Sep 17 11:35 Data
```



# Sharing with a group

By default, the group of a newly created file is the creator's primary group.

```
[dfr@lm4-f001 Data]$ touch testone  
[dfr@lm4-f001 Data]$ ls -l testone
```

Unless `newgrp` is used to change the group for the current session:

```
[dfr@lm4-f001 Data]$ newgrp adminucl  
[...]  
[dfr@lm4-f001 Data]$ touch testtwo  
[dfr@lm4-f001 Data]$ ls -l testtwo  
-rw-rw---- 1 dfr adminucl 0 Sep 18 10:45 testtwo  
[dfr@lm4-f001 Data]$ exit
```

# Sharing with a group

By default, the group of a newly created file is the creator's primary group.

```
[dfr@lm4-f001 Data]$ touch testone  
[dfr@lm4-f001 Data]$ ls -l testone
```

or the parent directory has `sgid` permission bit set:

```
[dfr@lm4-f001 Data]$ ls -ld .  
drwxrwx--- 4 dfr adminucl 9 Sep 18 10:45 .  
[dfr@lm4-f001 Data]$ chmod g+s .  
[dfr@lm4-f001 Data]$ ls -ld .  
drwxrws--- 4 dfr adminucl 9 Sep 18 10:45 .  
[dfr@lm4-f001 Data]$ touch testthree  
[dfr@lm4-f001 Data]$ ls -l testthree  
-rw-rw---- 1 dfr adminucl 0 Sep 18 10:48 testthree
```

# Sharing and hiding

When a common group is not available for sharing, the file can be *world-readable* in a *non-readable* but *traversable* directory. Then only users who know about the file exact name can open it.

```
[dfr@lm4-f001 ~]$ chmod o+x Download
[dfr@lm4-f001 Downloads]$ namei -l $(realpath rqlite-v7.21.1-linux-amd64.tar.gz)
f: /home/users/d/f/dfr/Downloads/rqlite-v7.21.1-linux-amd64.tar.gz
dr-xr-xr-x root root /
drwxr-xr-x root root home
drwxr-xr-x root root users
drwxr-xr-x root root d
drwxr-xr-x root root f
drwxr-x--x dfr dfr dfr
drwxrwx--x dfr dfr Downloads
-rw-rw-r-- dfr dfr rqlite-v7.21.1-linux-amd64.tar.gz
```

```
[bvr@lm4-f001 ~]$ ls ~dfr/Downloads/
ls: cannot open directory '/home/ucl/pan/dfr/Downloads/': Permission denied
[bvr@lm4-f001 ~]$ ls ~dfr/Downloads/rqlite-v7.21.1-linux-amd64.tar.gz
/home/ucl/pan/dfr/Downloads/rqlite-v7.21.1-linux-amd64.tar.gz
[bvr@lm4-f001 ~]$ file ~dfr/Downloads/rqlite-v7.21.1-linux-amd64.tar.gz
/home/ucl/pan/dfr/Downloads/rqlite-v7.21.1-linux-amd64.tar.gz: gzip compressed data, from Unix, original size 39096320
```

# Sharing and encrypting

The `gocryptfs` tool makes the process easy.

## 1. Install it

```
wget https://github.com/rfjakob/gocryptfs/releases/download/v2.4.0/gocryptfs_v2.4.0_linux-static_amd64.tar.gz
tar xvzf gocryptfs_v2.4.0_linux-static_amd64.tar.gz
chmod +x gocryptfs
mv gocryptfs [some directory in your PATH]
```

## 2. Create a directory that will contain the encrypted files and initialise a *vault*

```
[dfr@lm4-f001 ~]$ mkdir $CECIHOME/SecretFolder
[dfr@lm4-f001 ~]$ gocryptfs -init $CECIHOME/SecretFolder
Choose a password for protecting your files.
Password:
Repeat:
```

Your master key is:

```
1a88a6b1-8f072fe8-7aac5356-1d025115-
7574f7c3-627cbbdb-12b96ca8-09bfb39a
```

If the `gocryptfs.conf` file becomes corrupted or you ever forget your password, there is only one hope for recovery: The master key. Print it to a piece of paper and store it in a drawer. This message is only printed once.

The `gocryptfs` filesystem has been created successfully.

You can now mount it using: `gocryptfs /CECI/home/ucl/pan/dfr/SecretFolder MOUNTPOINT`

# Sharing and encrypting

## 3. *mount* the vault in a temporary directory

```
[dfr@lm4-f001 ~]$ gocryptfs $CECIHOME/SecretFolder ./Tests/ClearFolder
Password:
Decrypting master key
Filesystem mounted and ready.
```

## 4. Write files to the temporary directory

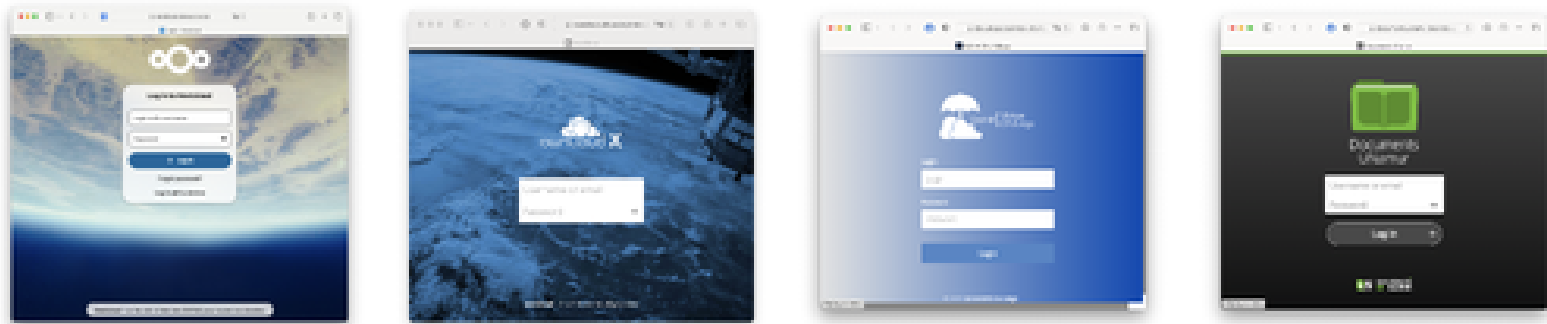
```
[dfr@lm4-f001 ~]$ echo test > ./Tests/ClearFolder/test.txt
[dfr@lm4-f001 ~]$ ls ./Tests/ClearFolder
test.txt
[dfr@lm4-f001 ~]$ ls $CECIHOME/SecretFolder
e6AxIMr4RuztuwpA-o_u0Q  gocryptfs.conf  gocryptfs.diriv
```

The files are encrypted on the fly.

Cleanup with `fusermount -u ./Tests/ClearFolder`

# Sharing with external colleagues

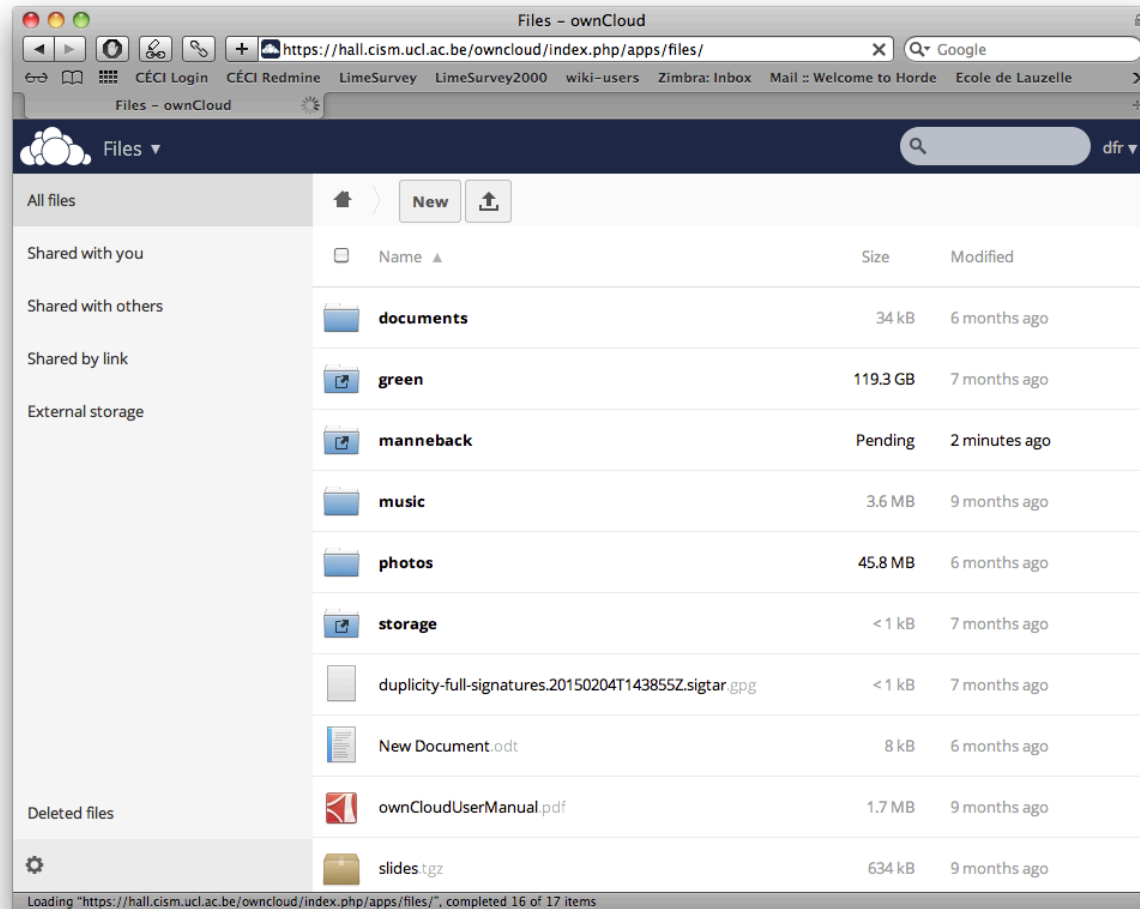
## Private cloud



## Open data

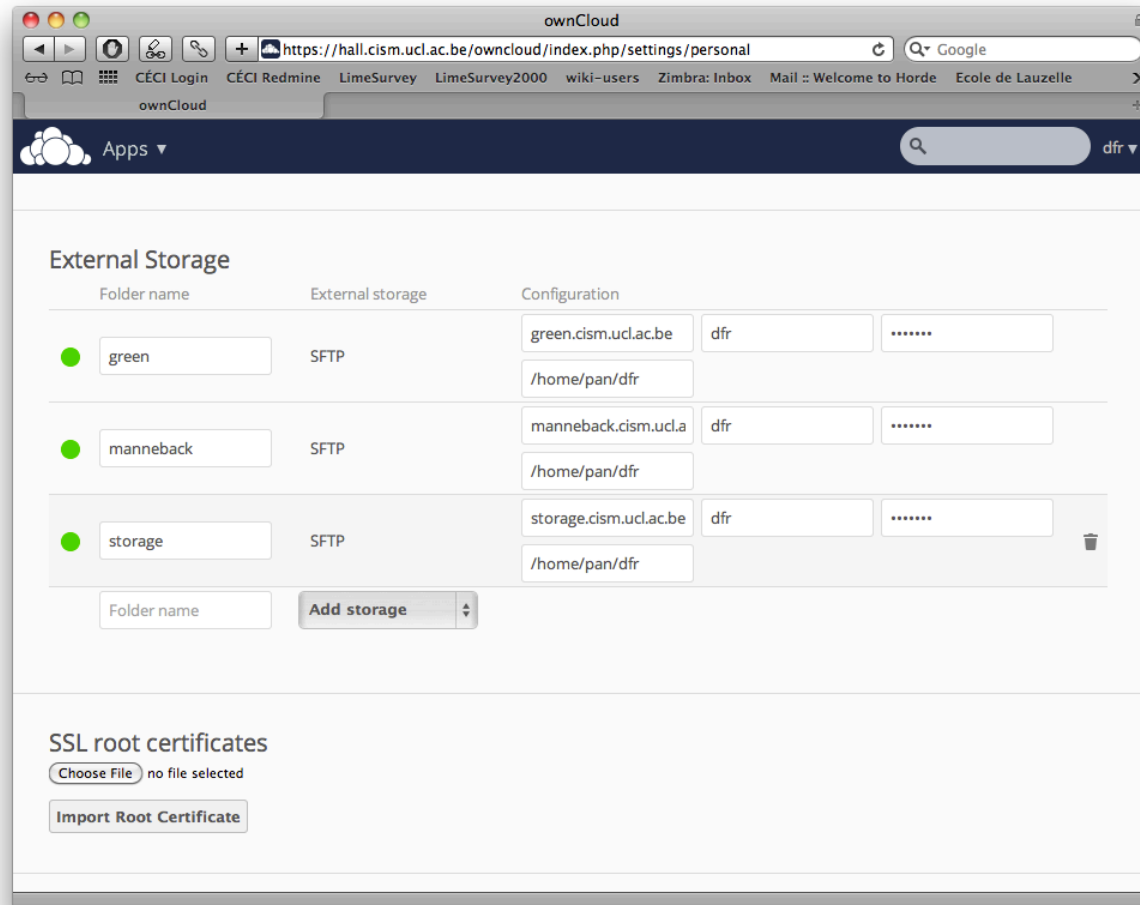


# Private cloud



Similar to Dropbox, OneDrive, Google Drive, etc.

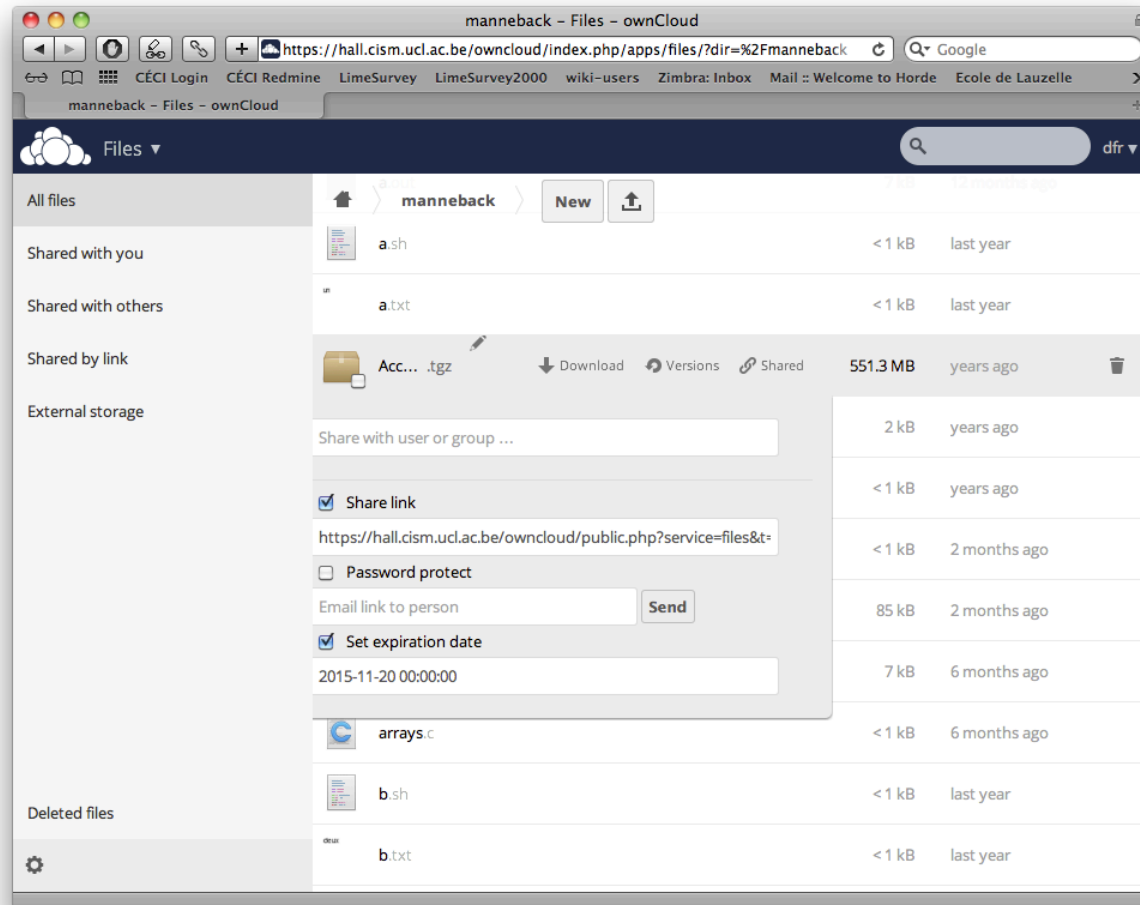
# Private cloud



Possibility to connect external storage

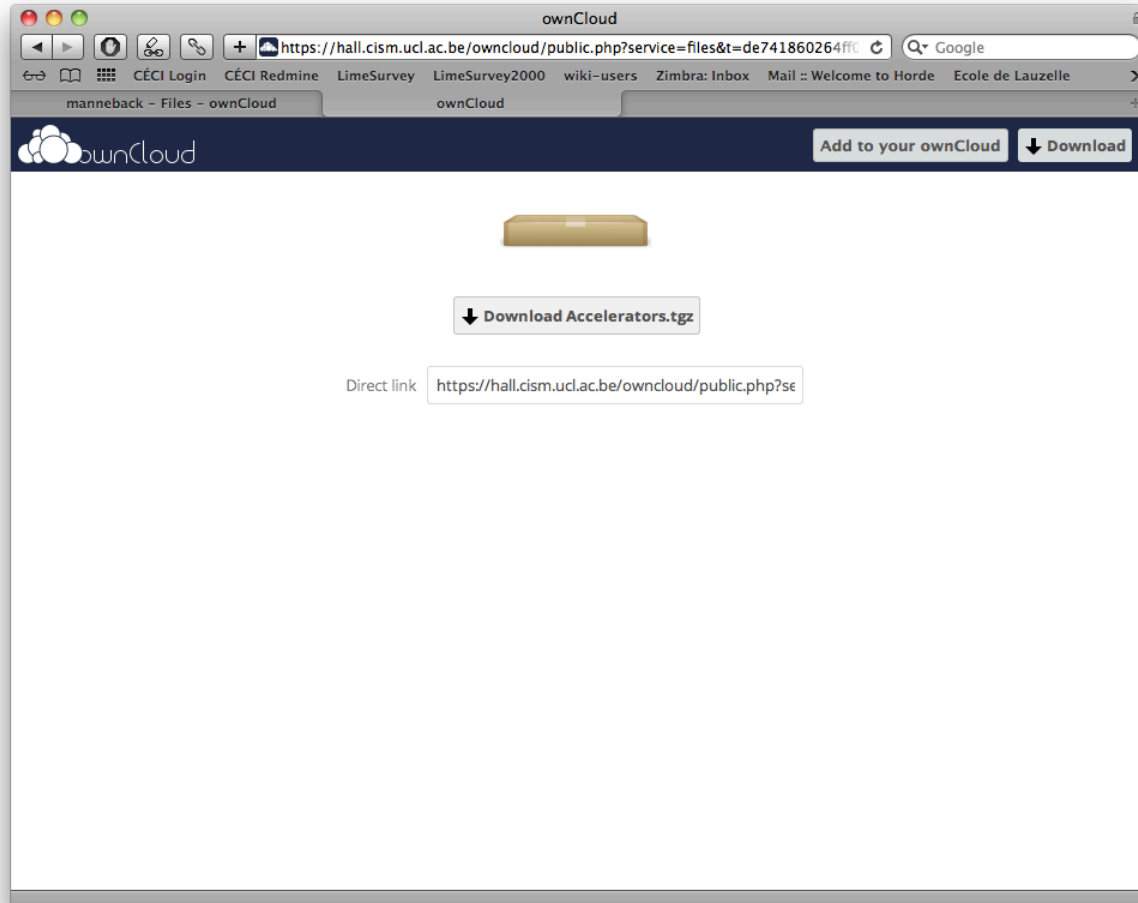


# Private cloud



Share with a download link

# Private cloud



Share with a download link

# Open data

The screenshot shows a web browser window displaying the 'Open Data @ UCLouvain' page on the Dataverse platform. The browser's address bar shows 'dataverse.uclouvain.be'. The page header includes the 'Dataverse' logo and navigation links for 'Search', 'User Guide', 'Support', 'Sign Up', and 'Log In'. The main content area features the 'UCLouvain' logo and the text 'Open Data @ UCLouvain (uclouvain) Université catholique de Louvain'. Below this, there is a 'Metrics' section showing '6 Downloads' and links for 'Contact' and 'Share'. A carousel of logos for various departments is displayed, including 'Dataverse of CISM', 'Dataverse of ELI', 'Dataverse of ICTEAM', and 'Dataverse of IMCN'. A search bar is present with the text 'Search this dataverse...' and buttons for 'Find' and 'Advanced Search'. The search results section shows '1 to 10 of 12 Results' and a 'Sort' dropdown. The first result is 'Dataverse of SCEB (uclouvain)' dated 'Oct 7, 2019'. The second result is 'Dataverse of CP3 (IRMP)' dated 'Sep 5, 2019', with a description: 'The UCL Centre for Cosmology, Particle Physics and Phenomenology (CP3) hosts research on high energy particle physics, cosmology, phenomenology and theory of the fundamental interactions. It is strong on both the experimental and theoretical fronts. The aim of the Centre is to br...'. The left sidebar contains filters for 'Dataverses (9)', 'Datasets (3)', and 'Files (3)', along with 'Dataverse Category' (Department (7), Research Group (2)), 'Publication Year' (2019 (12)), and 'Author Name'. A 'Display a menu' button is located at the bottom left of the sidebar.

Publish and reference your data with a DOI

# Summary and recap

## Summary and recap

**Storage:** choose the right file system and the right file format and give other storage systems some consideration

**Transfer:** use rsync in parallel when you can

**Sharing:** use all the potential of the UNIX permissions and try Nextcloud and Dataverse