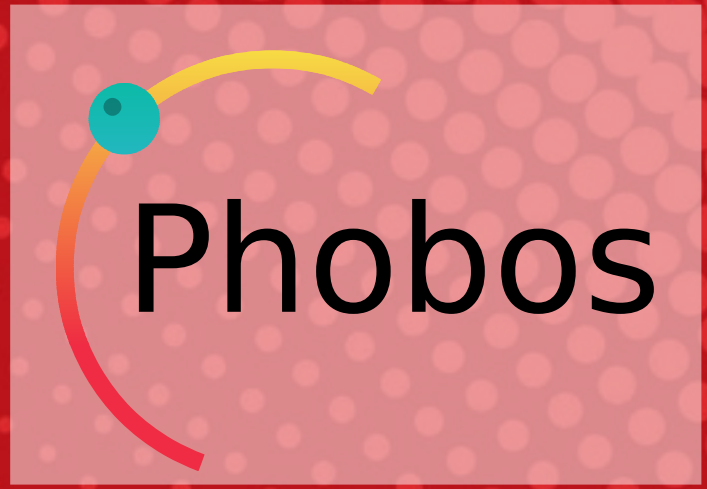


Unlock your data with **Phobos**

Thomas LEIBOVICI, thomas.leibovici@cea.fr

Long-term storage of huge amounts of data

Daily increase

Production of HPC systems:

- Today: 100s of TB
- Tomorrow: Petabytes

Accumulation over time

- Today: 100s of PB
- Tomorrow: Exabytes

Source: http://www.clker.com

For decades…

Disk vendors

“Tape will die”

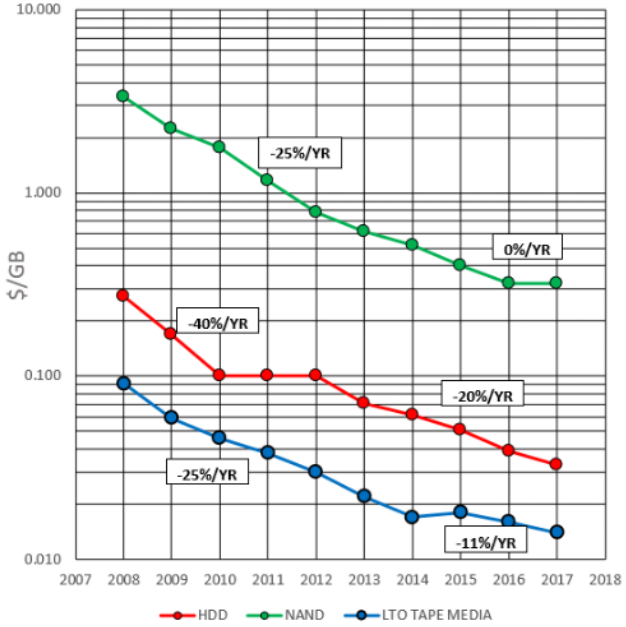
Tape vendors

“Disk will die”

Source: https://www.altaro.com

In Fact…

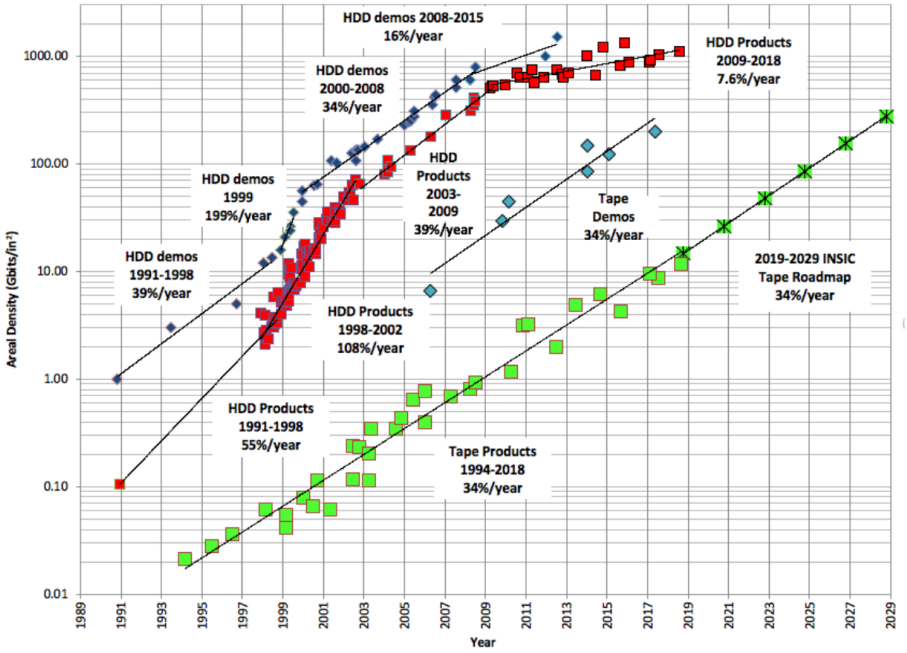
Tape remains the cheapest technology to store data



Source: CERN

Tape is not dead

And the technology keeps evolving



Source: INSIC

Existing solutions for scalable tape storage

Drawbacks of existing solutions:

- Vendor lock-in
  - Proprietary code, formats and protocols
  - Lacking integration to standards
- Expensive
  - Licenses
  - Complex (need local expertise)
- Provide much more feature than needed (=> complex and expensive)
- Heavy installation and maintenance operations

Fortunately, there is a great software…

LTFS

Linear Tape File System

https://github.com/LinearTapeFileSystem/ltfs

- Open-source
  - Main contributor: IBM
- Standardized format (ISO/IEC 20919:2021)
- Provides easy and efficient access to tapes
  - But no library control

Idea: “let's wrap it into a parallel storage system”

It should be easy, let's implement a software that:

1) Loads a free tape into a drive

2) Use LTFS to store files on tape

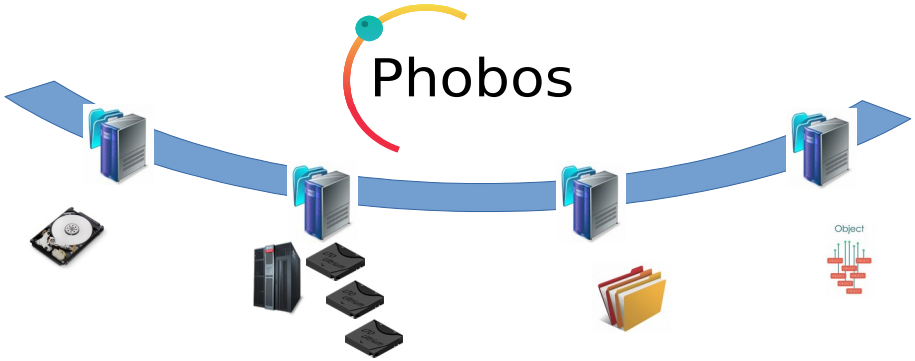
3) Remember on which tape the file is stored (e.g. in a database)

But… wait…

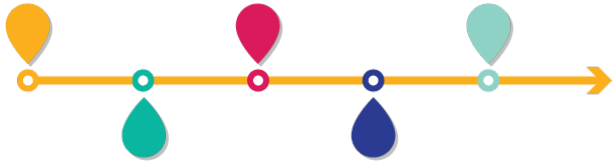
What about:
- Parallelizing accesses on multiple servers?
- Object consistency in case of current accesses?
- Object versioning?
- Object deletion?!
- Mirroring / Erasure coding?
- Managing different generations of tapes
- Manage tape life cycle
- Detect faulty devices or media
- Managing a disk cache
…

Phobos: Parallel Heterogeneous Object Store

- Goals :
  - Manage a distributed set of storage resources on various storage technologies (HDD, tapes, object stores…)
  - Implement the best I/O optimizations for each technology without compromise
    - E.g. for tapes: minimize mounts and data sync



History of the Project



- 2013: first ideas
- 2014-2015: development of the initial version
  Scope:
  - Storage on tape using LTFS, or in a filesystem
  - SCSI-controlled tape library and LTO drives
  - Single server
- 2016: **Phobos in production**
  - Multi-Petabyte storage of genomics data
  - IBM TS3500 library, LTO5/6 drives

FRANCE GÉNOMIQUE

- 2019: Phobos made **open-source** (LGPL v2.1), available on github
- 2020-2022: Towards Phobos 2.0 → Parallelizing Phobos
- September 2022: First **parallel** version of Phobos **in production** as **Lustre/HSM backend**

Guidelines

Design guidelines

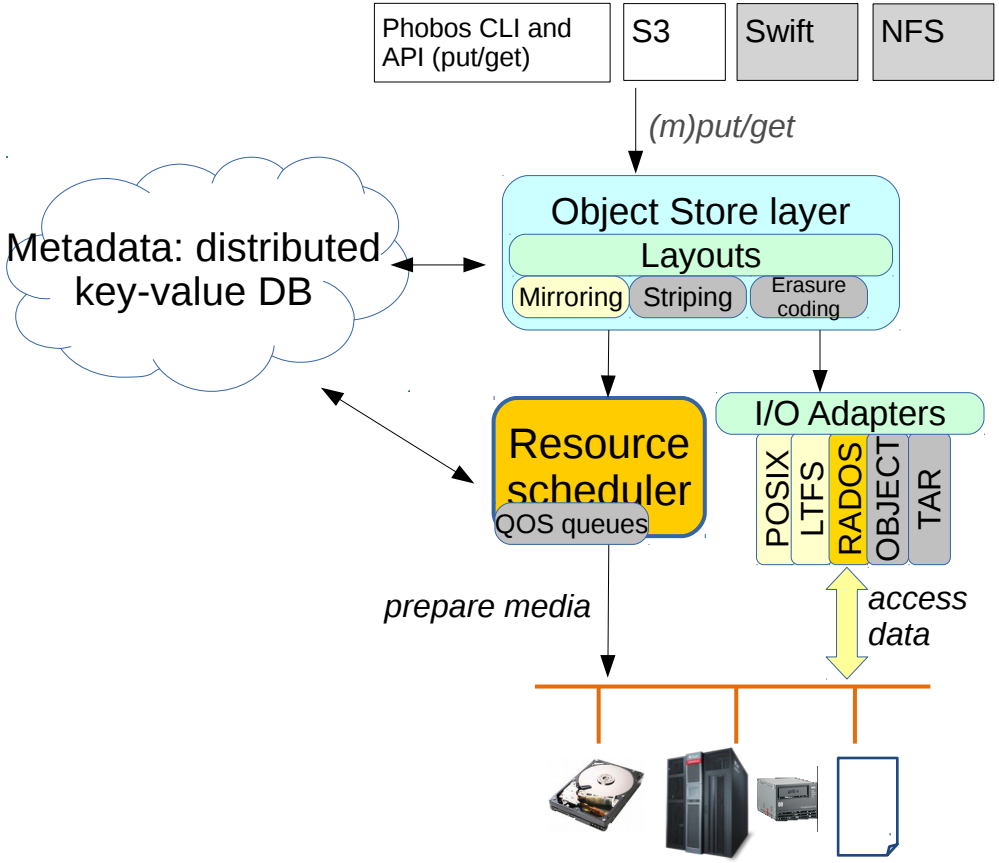
- Scalability and fault-tolerance
- Based on open formats, open protocols, interoperable
  - E.g. LTFS as tape filesystem (ISO/IEC 20919:2016)
- Simple and common interfaces (CRUD API, REST)
- Simple administration (intuitive, admin-friendly CLI)
- Light, easy to deploy, easy to maintain
  - As of today: 48k lines of C and Python

Coding guidelines

- State of the art of code quality: 2 reviews+gatekeeping, unit tests, integration tests, system tests, static and dynamic code checks…

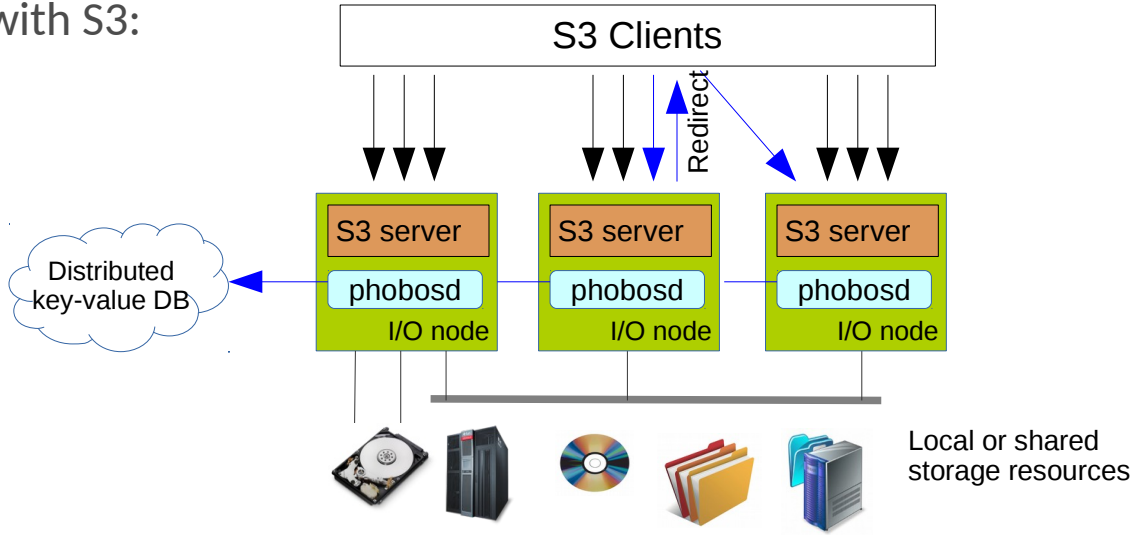
Phobos components overview

- **IO adapters**: support of multiple storage backends (Posix, LTFS, RADOS)
- **Layout plugins**: performance and fault-tolerance
- **Resource scheduling**: optimizes stream to tape drives, minimizes tapes mounts
- **Front-ends**: CLI, API, S3, more to come
- **Key-value metadata schema**:
  - DB schema is NoSQL-ready
  - Currently uses PostgreSQL: can be parallelized thanks to sharding features
  - Backup copy of metadata stored with objects (recovery, tape import)



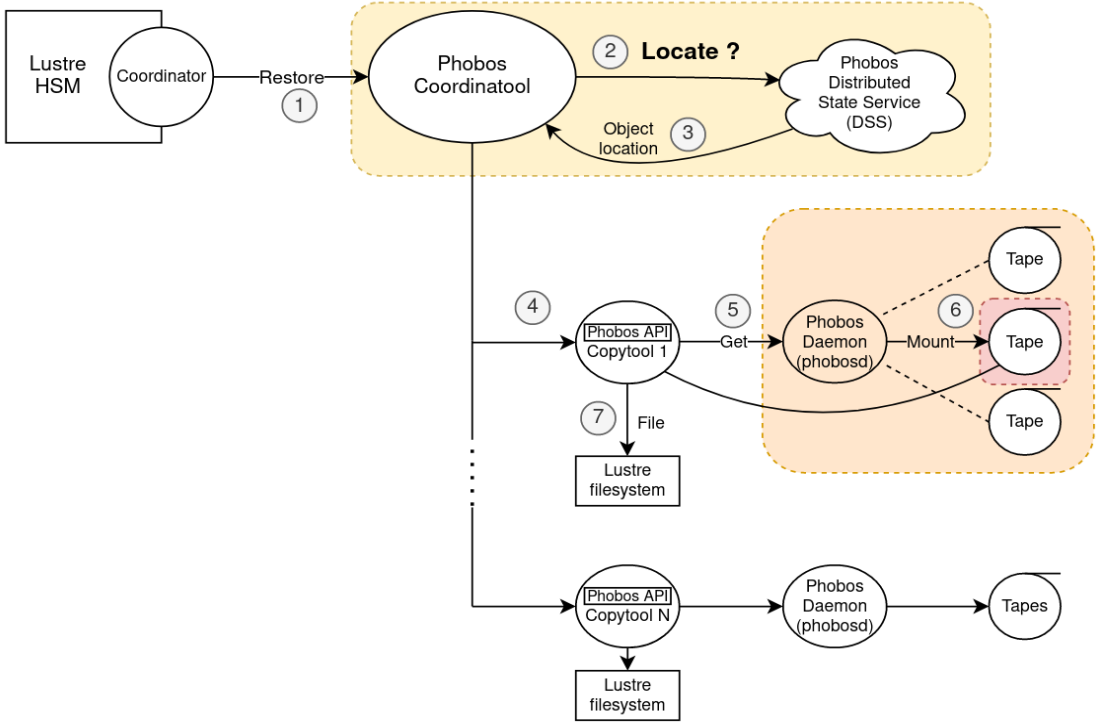
Distributing Phobos

- Phobos can run on multiple servers, with a shared database
- I/O distribution relies on the “phobos_locate” feature to direct the I/O to the right Phobos server
- The use of this feature is up to the Front-end
- Example with S3:



Distributing Phobos

- Other example with Lustre/HSM:



Phobos features illustrated

Easy setup

- Drive setup

```
phobos drive add --unlock /dev/st1
```

- Tape addition & formatting:

```
phobos tape add -t lto6 [073200-073222]L6

phobos tape format --nb-streams 3 --unlock [073200-073222]L6
```

All done! Phobos is ready for I/Os!

Phobos features illustrated

Resource partitioning with tags

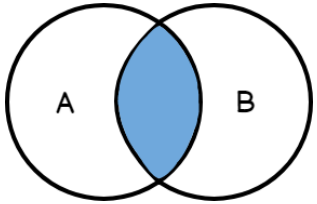
- Tagging resources

```
phobos tape update --tags project1,classB [073000-073099]L8
```

- Pushing data to specific resources:

```
# push data to any media with tag “classB”
phobos put --tags classB /path/to/file  objid1

# push data to a media with both tags “project1” and “classB”
phobos put --tags project1,classB /path/to/file objid2
```



Phobos features illustrated

Object versioning

- Object uniqueness

```
phobos put /path/to/file objid1
```

→ fails if *objid1* exists

- Creating new object version

```
phobos put --overwrite /path/to/file objid1
```

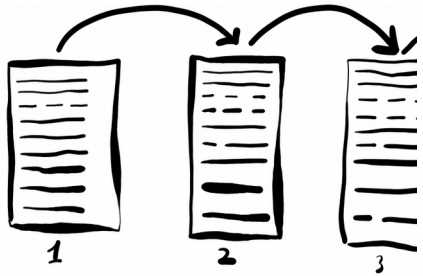
→ creates a new version of *objid1*

- Listing object versions

```
phobos object list --deprecated objid1
```

- Retrieving an old version

```
phobos get --version 1 objid1 file.out
```



Phobos features illustrated

Object deletion

- Deletion

```
phobos del objid1
```

- Cancelling deletion

```
phobos undel objid1
```

- Listing deleted versions

```
phobos object list --deprecated objid1
```

- Retrieving a deleted version

```
phobos get --uuid ABC12312 --version 2 objid1
```

Available until the media is “repacked”

Phobos features illustrated

Arbitrary attributes

- Attaching arbitrary attributes to objects

```
phobos put /path/to/file objid1 --metadata \
                                “cksum=md5:7c28...5e3e,user=foo”
```

- Querying

```
phobos getmd objid1
```

- Filtering

```
phobos object list --metadata “user=foo” “obj*”
```

```
| oid   | user_md                                      |
|-------|----------------------------------------------|
| obj01 | {"user": "foo", "crtime" : "132948897"}|
| obj02 | {"user": "foo"}                              |
```

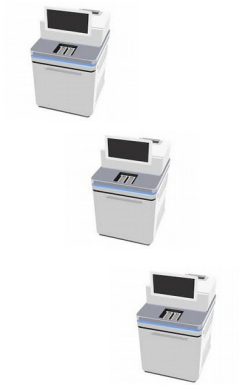
Example of deployments

In production use-case

FRANCE GÉNOMIQUE

- Multi-petabyte genomics datasets
- In production since 2016

DNA sequencers



Phobos

- IBM TS3500 tape library (SCSI)
- LTO6 and LTO8 drives

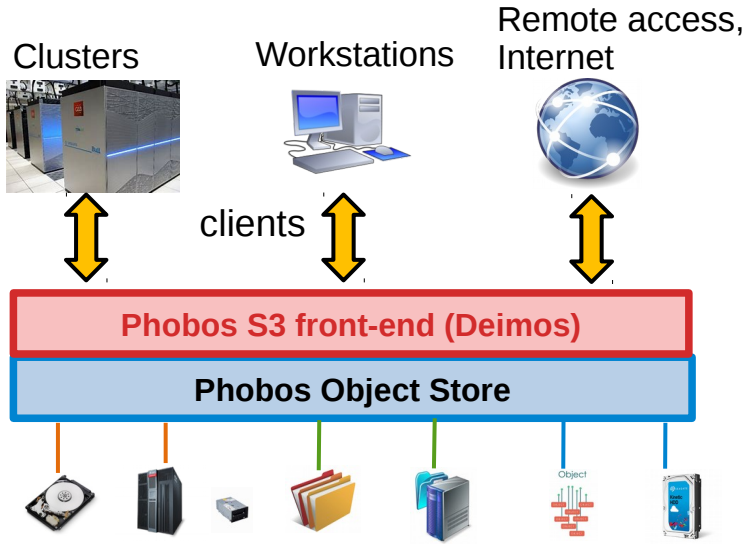
HPC data clusters

TGCC
Très Grand Centre de calcul du CEA

Object store with an S3 front-end

- S3 interface exposed to end-users
- Phobos: high-performance, scalable storage
  - Can manage a wide variety of high-capacity storage, including tape libraries
  - Provides an easy/uniform management of the storage resources

- Phobos' S3 front-end developed by ICHEC: https://git.ichec.ie/performance/storage/deimos

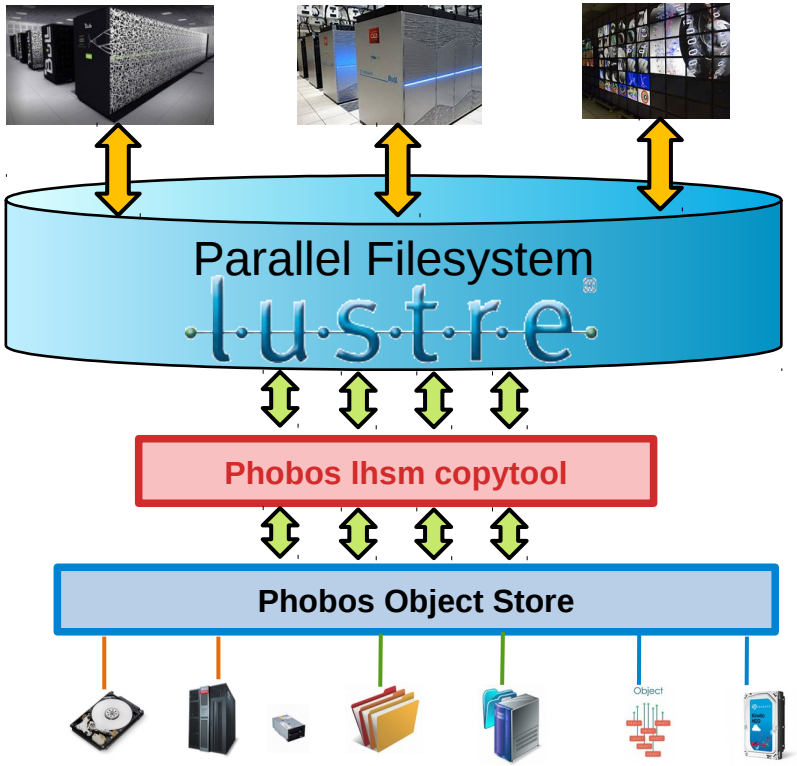


Lustre/HSM backend

Lustre HSM backend

- Lustre: filesystem user front-end
- Phobos as high-capacity backend (hierarchical storage)

- **In production this year at CEA**



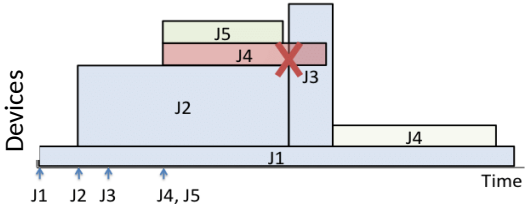
Community

- Collaboration with DDN and ICHEC:
  - Implementing a S3 server for Phobos: Deimos
  - Contribution to Phobos: “alias” feature

- Collaboration with Atos, ECMWF, ICHEC, Seagate, Univ. of Mainz
  - In the framework of the EuroHPC project “IO-SEA”
  - Building a storage software stack for Exascale systems
  - Phobos used as the long-term storage component
  - New developments: scalability enhancements, erasure coding, media lifecycle management, administrative interface, LTFS tape import, smart tape request reordering, front-ends (Swift, POSIX)...

Roadmap

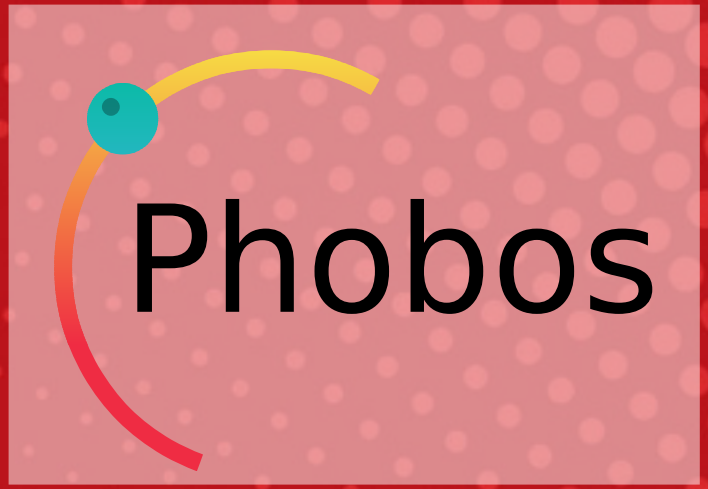
- Current development: optimized I/O scheduling for tapes
  - Short term focus on grouping I/Os on tapes
  - Still much to do (local IO scheduling, global IO strategy, organization of device utilization over time...)
- 2H 2022: media life cycle (policy-based repacks...)
- 1H 2023:
  - internal data migration (policy-based)
  - NFS front-end
- Other planned enhancements:
  - Disaster recovery
  - Media import
  - New layouts (e.g. erasure coding)
  - New front-ends / new backends



Interested?

- Start here: https://github.com/phobos-storage
- Contributions are welcome, as well as feedback!

DE LA RECHERCHE À L'INDUSTRIE



https://github.com/phobos-storage

Thank you for your attention!