# Phobos tutorial

Guillaume Courrier
Sebastien Gougeaud

December 2, 2022

This document has been written and tested with Phobos 1.94.1. It assumes that the reader has access to a VM with Phobos installed and QuadstorVTL to manage tapes. Each exercise assumes that the previous ones were done.

## 1 Basic commands

### 1.1 Setup

Before we can use Phobos, we need to setup the database. Phobos provides a tool `phobos_db` for this purpose:

```
sudo -u postgres phobos_db setup_db
```

You will be asked to input a password for the database access. This password should match the one in the configuration `/etc/phobos.conf`. The option `connect_string` contains the information necessary for the connection to the PostgreSQL database. Make sure that the password matches as well as the host which should be `localhost` for this tutorial.

Once the database is setup, we can initialize the tables:

```
sudo phobos_db setup_tables
```

We can see that everything is setup properly by listing the objects in the database:

```
phobos object list
```

Since the database is empty, we should not see any output. If the command fails, the database is not properly setup.

### 1.2 Managing Phobos resources

Before we can write data, we need to have somewhere to write. In Phobos, that somewhere is referred to as a **medium**. A medium can only be accessed though a `device`.

Phobos can manage three types of resources (i.e. medium):

**dir** any POSIX filesystem;

**tape** magnetic tapes inside a library;

**rados_pool** a pool of RADOS objects inside a Ceph cluster.

In this tutorial, we will not use RADOS pools since their management is very similar to POSIX directories and require some knowledge of Ceph.

## 1.3 POSIX Directories

The goal now is to write a file in a directory. For that, we want to execute the following command:

```
phobos put /etc/hosts my_first_object
```

1. Execute the command above.
2. What error message did you get? Why?
3. What process do you need to start?
4. Does the command above work now?
5. Where can the object be written in the current setup? Try to list all the medium currently available to Phobos.
6. What command allows you to manage POSIX directories? List all the available actions on POSIX directories.
7. In what state should a medium be in order to be written to?
8. Use the appropriate commands to setup a directory named `/tmp/archive`.
9. Does the put command work now? If not, which option of `phobos put` should you use to specify where the object should be written?
10. Open Phobos' configuration `/etc/phobos.conf`. Which option can you modify to write objects on directories by default? Try to put an object without the command line option.
11. Try to list all the information about the object written.
12. Read the content of the object into a file.
13. Given the output of `phobos extent list -o all`, try to find where the object was written in `/tmp/archive`. You can compare the content of the file to the original file.
14. Write an object with the metadata `foo=bar`. How can you list the metadata of this object ? List objects with the metadata `foo=bar`.

## 1.4 Linear Tapes

We will now do the same thing but on tape. At this point, you should have Phobos properly setup for writing to a directory. You will learn how to setup tapes in this section.

1. What physical element do you need to write data on tape?
2. To what Phobos command do they correspond to? List all the available options for these commands.
3. Using `lsscsi`, list all the available tape drives and their LTO generation.

4. Using `mtx`, list all the available drives and tapes
5. Using `mtx`, load a tape into a drive. Can you load an LTO5 tape into an LTO6 drive? An LTO6 tape into an LTO5 drive?
6. Add a drive to Phobos using the SCSI Tape device path. You can also try with the serial number (see `tapeinfo`).
7. Add a tape to Phobos compatible with the drive.
8. Try to format the tape. If this didn't work, in which state should the drive be?
9. Can you write an object on the tape now?
10. Check the state of the drive using `mtx`.
11. Repeat steps 11 and 12 from the previous exercise.
12. What do the fields displayed by `phobos drive status` mean?
13. Using the commands `mtx`, `lsscsi` and `tapeinfo`, try to find the information given by `phobos drive status`.
14. Given the output of `phobos extent list -o all`, where is the object written?
15. Use `phobos mput` to write several objects in one command line.

## 1.5 Inspecting the state of the database

1. Connect to the Phobos database: `psql phobos phobos`.
2. List all the tables: `\dt`.
3. Which table contains the path to the data of an object on the medium?
4. How does Phobos know where an object should be accessed?
5. What is the use of the `lock` table?
6. What devices are stored in the database?
7. What media are stored in the database?
8. List which `phobos` command lists the content of each table.

END OF TUTO

If you try now, you will see that it fails because it cannot contact `phobosd`:

```
<ERROR> Cannot contact 'phobosd': will abort: Transport endpoint is not connected
```

This is because, `phobos` cannot manage the media and devices directly. This needs to be done by a central component: `phobosd`.

```
systemctl start phobosd
```

The daemon should be up and running. Use `phobos ping` to see if it is able to accept new requests.

If you try the `phobos put` command again, it will still not work but for a different reason:
`No space left on device`.
This error was sent by the daemon because it currently have no medium on which it can write data. Hence, no space left on device.

So if we want to write data, we need to add a directory to Phobos:

```
mkdir /tmp/archive
phobos dir add /tmp/archive
```

Once added, we can see the information inside the database:

```
phobos dir list -o all
```

Once in the database, we need to format the directory and unlock it:

```
phobos dir format /tmp/archive
phobos dir unlock /tmp/archive
```

The `phobos put` command will now work.

You can read back the data and check if the retrieved file match:

```
phobos get my_first_object /tmp/out
diff /etc/hosts /tmp/out
```

We can also see our first object in the database:

```
phobos object list -o all
```