leBLOB

S3 STORAGE FOR THE MASSES

Object storage on your own servers.

leBLOB is an S3-compatible object store. One command starts it on one machine, for any S3 client to use. It is designed to grow into a cluster across several locations.

```
# one command; the first start prints a key
$ leblob serve --data-dir /srv/leblob

# any S3 client, with that key
$ export AWS_ENDPOINT_URL=http://localhost:9000
$ aws s3 mb s3://archive
$ aws s3 cp backup.tar s3://archive/

# moving in, with an rclone remote for each
$ rclone sync --metadata minio:photos leblob:photos
```

Free software under the Apache 2.0 licence · git.sr.ht/~sfermigier/leblob

THE GAP

Self-hosted S3 needs somewhere to go

MinIO

Its community edition has been wound down. Binaries and container images can no longer be downloaded, so an upgrade means building it from source.

One machine

Most self-hosted object stores run on a single server. They need something that installs in a minute and runs for years.

Your data

Backups, photos, build artefacts: what a store says it has written must still be there after a power cut.

leBLOB starts where most installs are, on one machine, and **grows into a cluster without a migration.**

THE DESIGN

Immutable blobs, consistency in the metadata

An object



On disk

randomly named files, written once, synced, then renamed into place

Metadata

```
cat.jpg → [blob 1, blob 2, blob 3]
one SQLite commit makes it visible
```

A write is one commit

The bytes are durable before the commit, so a crash before it changes nothing anyone can see. Clients never see half an object.

Copies can't disagree

A blob never changes, so its copies need no quorum. A read moves to another copy at the same offset if a disk fails.

The same model for a cluster

Only the metadata store changes. Stored data is never rewritten on the way up.

TESTED WHERE DATA GETS LOST

Durability is measured, not assumed

0

acknowledged writes lost in 240 simulated power cuts, on ext4, XFS and btrfs, with one copy and with two.

Every

filesystem catches a build that skips one `fsync`. CI runs that control on every push.

349

tests of Ceph's `s3-tests` pass, and a change that loses one fails the build.

4

clients tested end to end over HTTP and HTTPS: boto3, the AWS CLI v1 and v2, rclone and s5cmd.

Power cuts are replayed at the block layer from a log of every write the disk saw. Dying, full and rotting disks are tested under a real server too.

ONE SERVER, SEVERAL DISKS

What it does today

S3

- Multipart uploads and copies
- Versioning and lifecycle rules
- Object tags and CORS
- Public-read buckets and objects
- Presigned URLs and form uploads
- Encryption with the client's key

Operations

- Copies across several disks
- Read failover, scrub and repair
- Metadata backup while running
- Prometheus metrics
- Keys scoped per bucket
- Upgrades tested from each release

Moving in is one command.

```
rclone sync --metadata \
  old:bucket leblob:bucket
```

Tested from MinIO and Garage: bytes, content types and user metadata arrive, multipart objects and odd keys included.

EARLY, AND OPEN

Try it on your own machine

```
# from source, with uv
$ git clone https://git.sr.ht/~sfermigier/leblob
$ cd leblob && uv run leblob serve --data-dir ./data

# or in a container
$ docker build -t leblob .
$ docker run -p 9000:9000 -v leblob-data:/data leblob
```

Code: git.sr.ht/~sfermigier/leblob

> **Now:** one server, one or several disks. Alpha: run it next to your backups first.

> **Next:** running it for real, for weeks, under real clients.

> **Then:** clusters, with several locations in mind from the start.

Written in Python, Apache 2.0. In France, "le blob" is a slime mould that grows networks between distant points.