Testing storage and metadata backends

Hugo González Labrador, Arno Formella

LIA2, University of Vigo

CS3: Cloud Storage Services for Novel Applications and Workflows
Zürich, January 2016

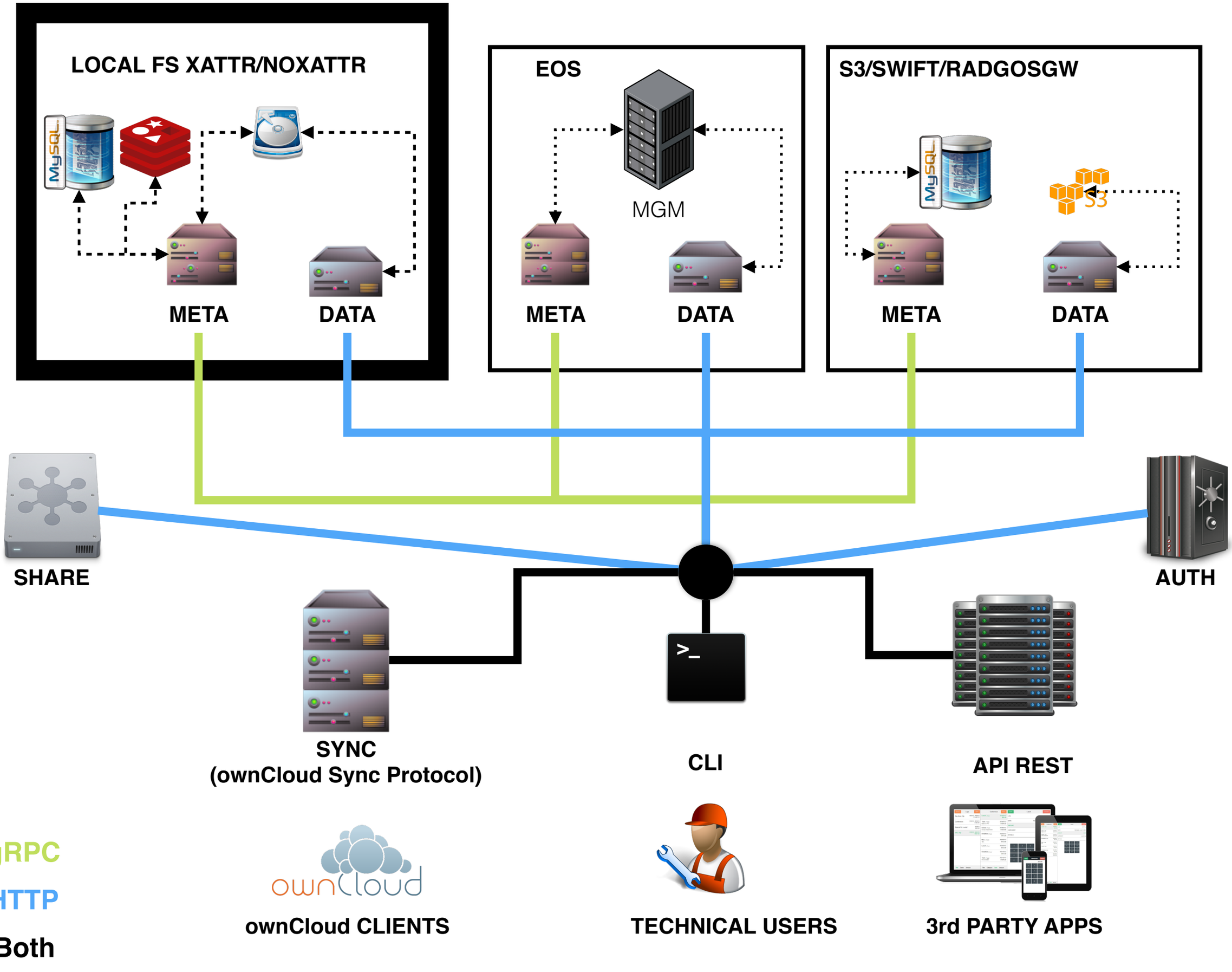
Outline

- Origin of the project
- Architecture
- Storage backends
- Benchmark results
- Conclusion
- Outlook

Why did we develop ClawIO ?

- **Cloud Synchronisation Benchmarking Framework**
- Curiosity for testing new data and metadata backends that are novel for synchronisation platforms.
- Flexible to plug your implementations in any language and technology
- Experiment with a new design that:
    - Avoids synchronisation between the DB (containing the metadata) and the filesystem (containing the data) that is done in the majority of sync platforms using a local filesystem.
- Experience from being a Technical student at CERN working on the CERNBox project.

Architecture



Metadata used by ownCloud

Data

Metadata Key	Metadata Value
CHECKSUM	md5:8c8d357b5e872bbacd45197626bd5759
MTIME	104857600
PATH	/local/users/d/demo/file
FILEID	a8584c90-ae2a-4dd8-84a7-f18ced109cce
ETAG	956494f8-5120-4165-afba-ad5f8d13b8ef

SETUP ONE: Local FS with MySQL as the metadata store

FILEID
CHECKSUM
MTIME
ETAG

Ext4 FS

PATH

/
/bin /dev /etc /private /sbin /tmp /usr
/usr/bin /usr/home /usr/lib /usr/local /usr/sbin /usr/share

DATA

META UNIT

DATA UNIT

SETUP TWO: Local FS with MySQL and XATTRs as metadata stores

~~FILEID~~
~~CHECKSUM~~
MTIME
ETAG

Ext4 FS XATTR

FILEID
CHECKSUM

PATH

/bin /dev /etc /private /sbin /tmp /usr

/usr/bin /usr/home /usr/lib /usr/local /usr/sbin /usr/share

DATA

META UNIT

DATA UNIT

SETUP THREE: Local FS with REDIS and XATTRs as metadata stores

~~FILEID~~
~~CHECKSUM~~
MTIME
ETAG

Ext4 FS XATTR

FILEID
CHECKSUM

/

PATH

/bin /dev /etc /private /sbin /tmp /usr

/usr/bin /usr/home /usr/lib /usr/local /usr/sbin /usr/share

DATA

META UNIT

DATA UNIT

VM Machine Specs, Deployment scenario and Operations to benchmark

VM

infrastructure provided by

CITI CENTRO DE INVESTIGACIÓN TRANSFERENCIA E INNOVACIÓN UNIVERSIDADE DE VIGO

CPU	**64 cores Intel(R) Xeon(R) CPU E5-4640 v2 @ 2.20GHz**
RAM	**64 GB**
DISK	**SAS-3 12 Gb/s 4 TB Seagate Constellation Enterprise ST4000NM3401 (RAID6)**

16 services (servers) on the same VM

Deployment

docker

bench client also on the same VM

Operations

Stat 

Upload 

STAT benchmark

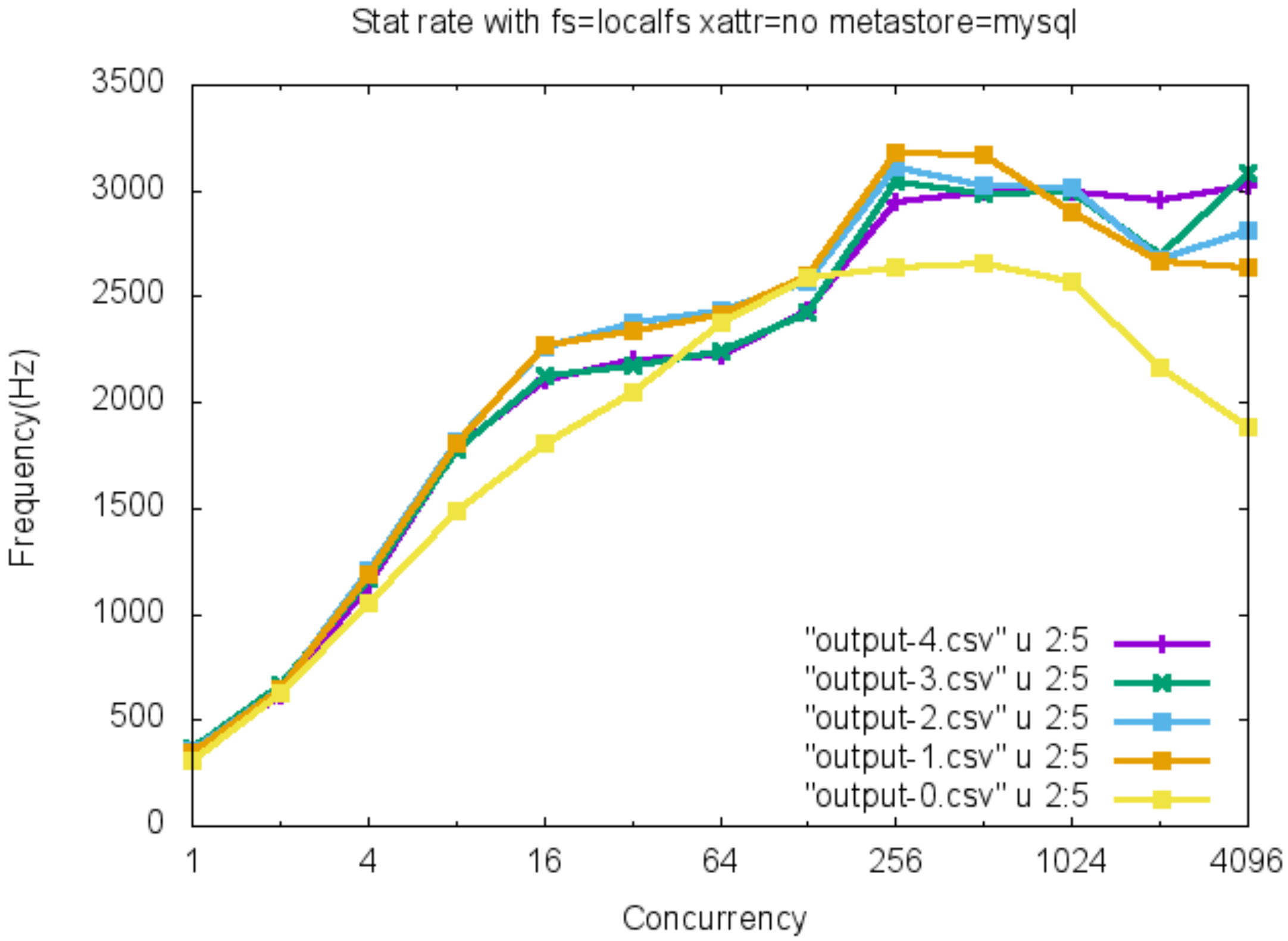
- The STAT operation is similar to a Unix file stat operation or a webDAV **PROPFIND**.

- The objective of this operation is to **retrieve the metadata** associated with a particular resource (file or folder)

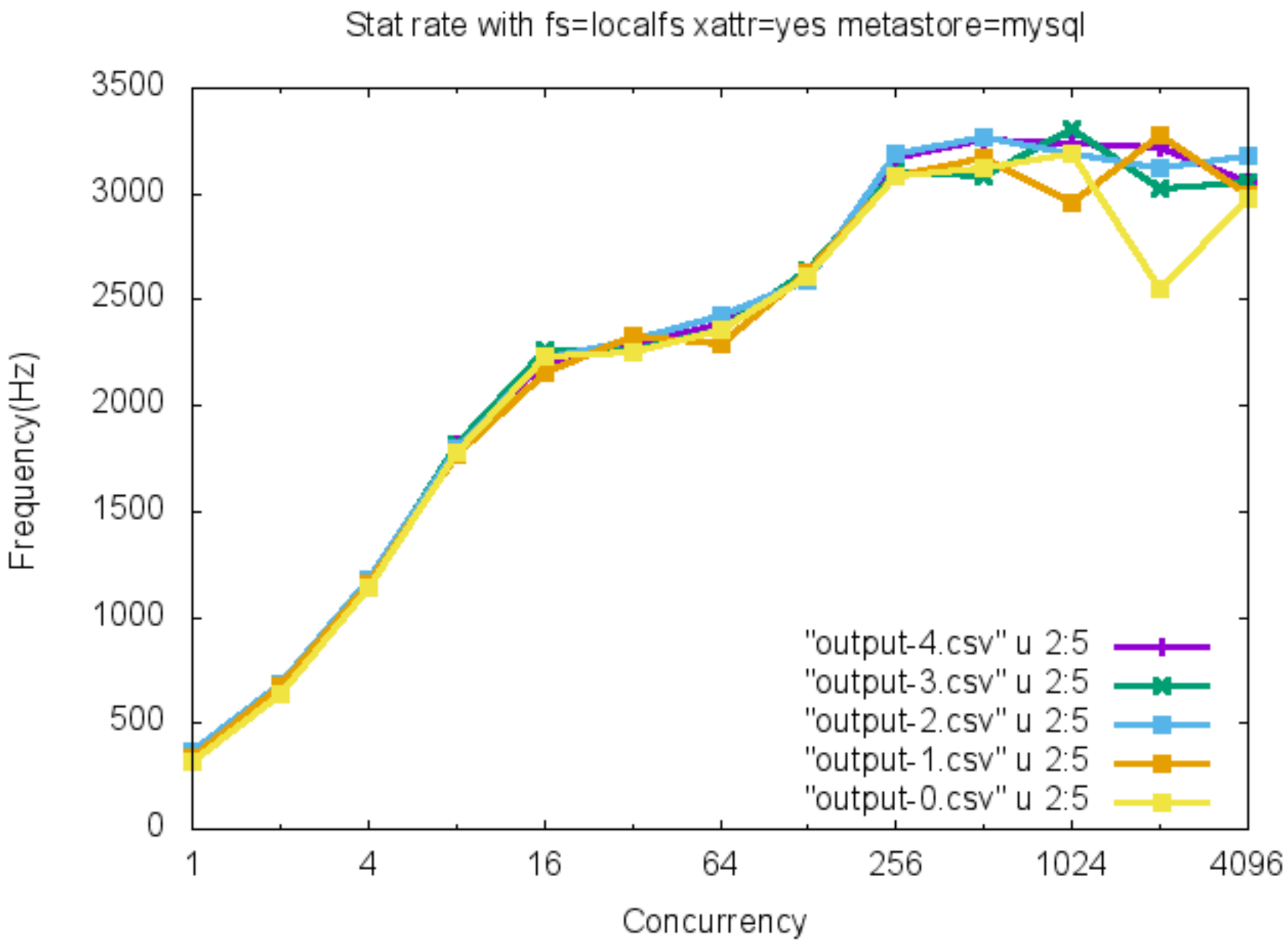
- For each level of concurrency, **10 000** requests are triggered and the test is **repeated 5 times.**

- Operation uses **gRPC** and **HTTP/2**.

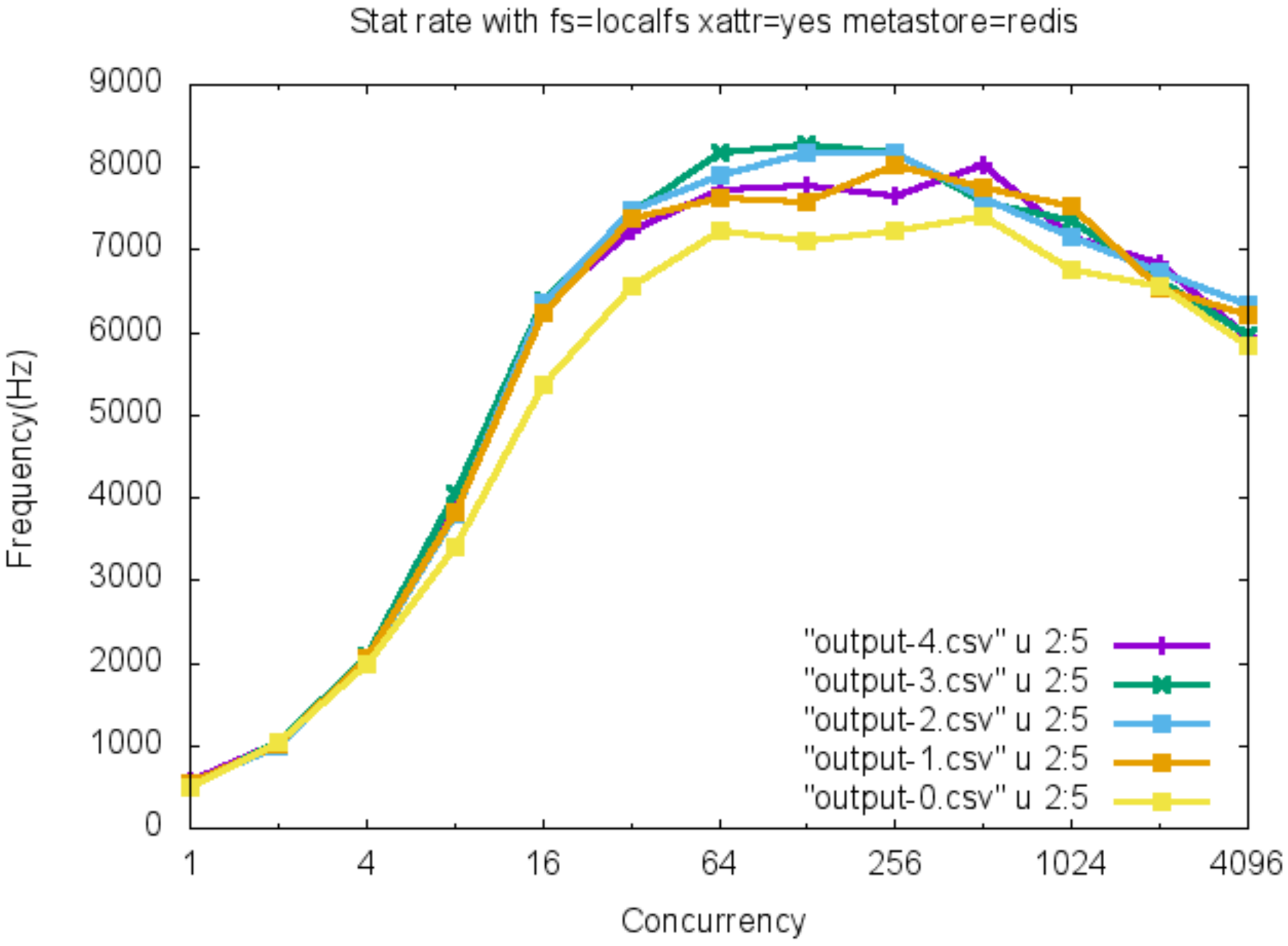
STAT benchmark



STAT benchmark



STAT benchmark



UPLOAD benchmark

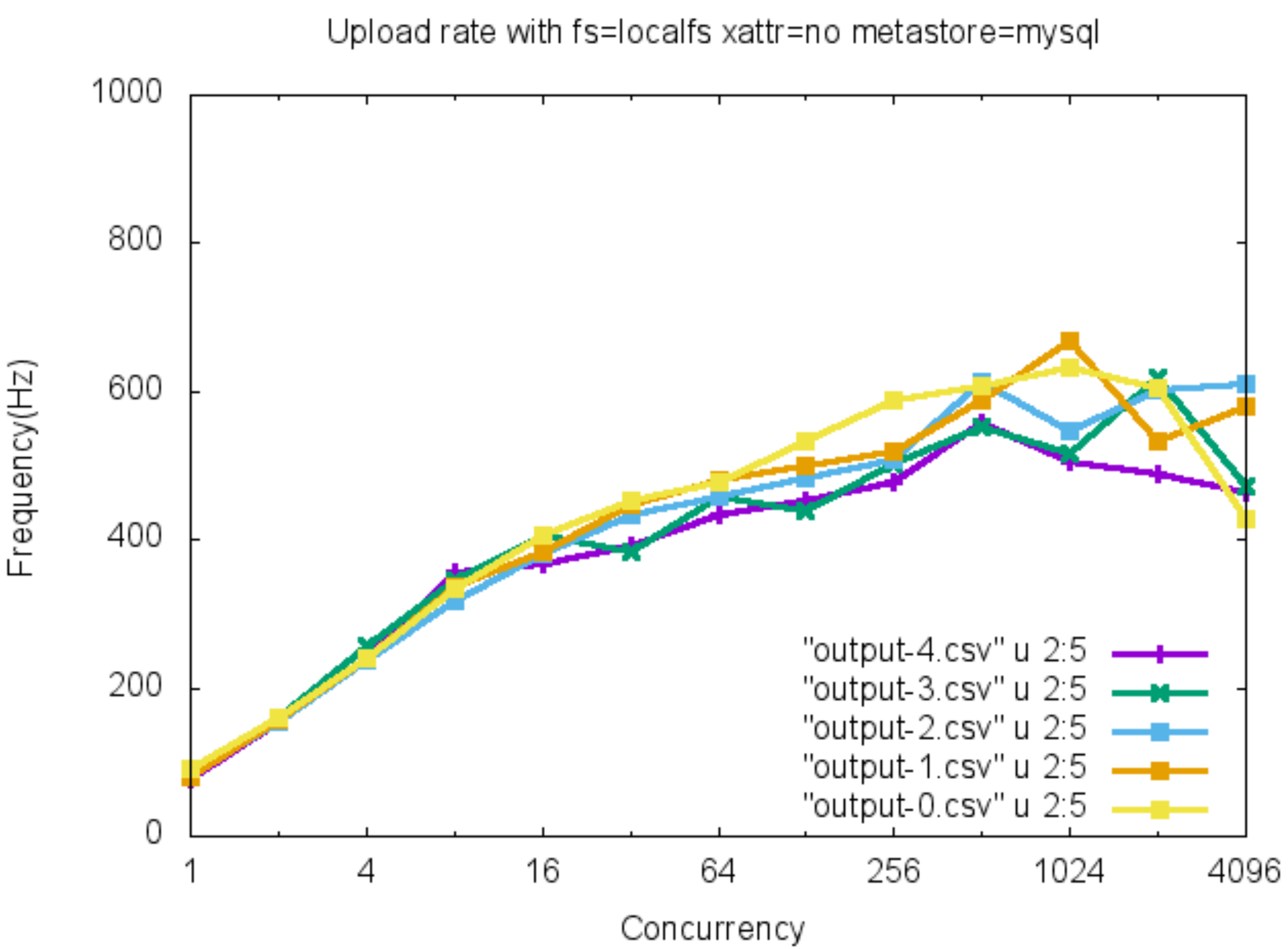
- The upload operation uploads a **file randomly** chosen from a fixed set of 100 files that follow the **distribution of files observed on CERNBox**.

- The chosen file is uploaded **5000** times per concurrency level, to **random target destinations** to avoid overwrites. The benchmark is **repeated 5 times**.

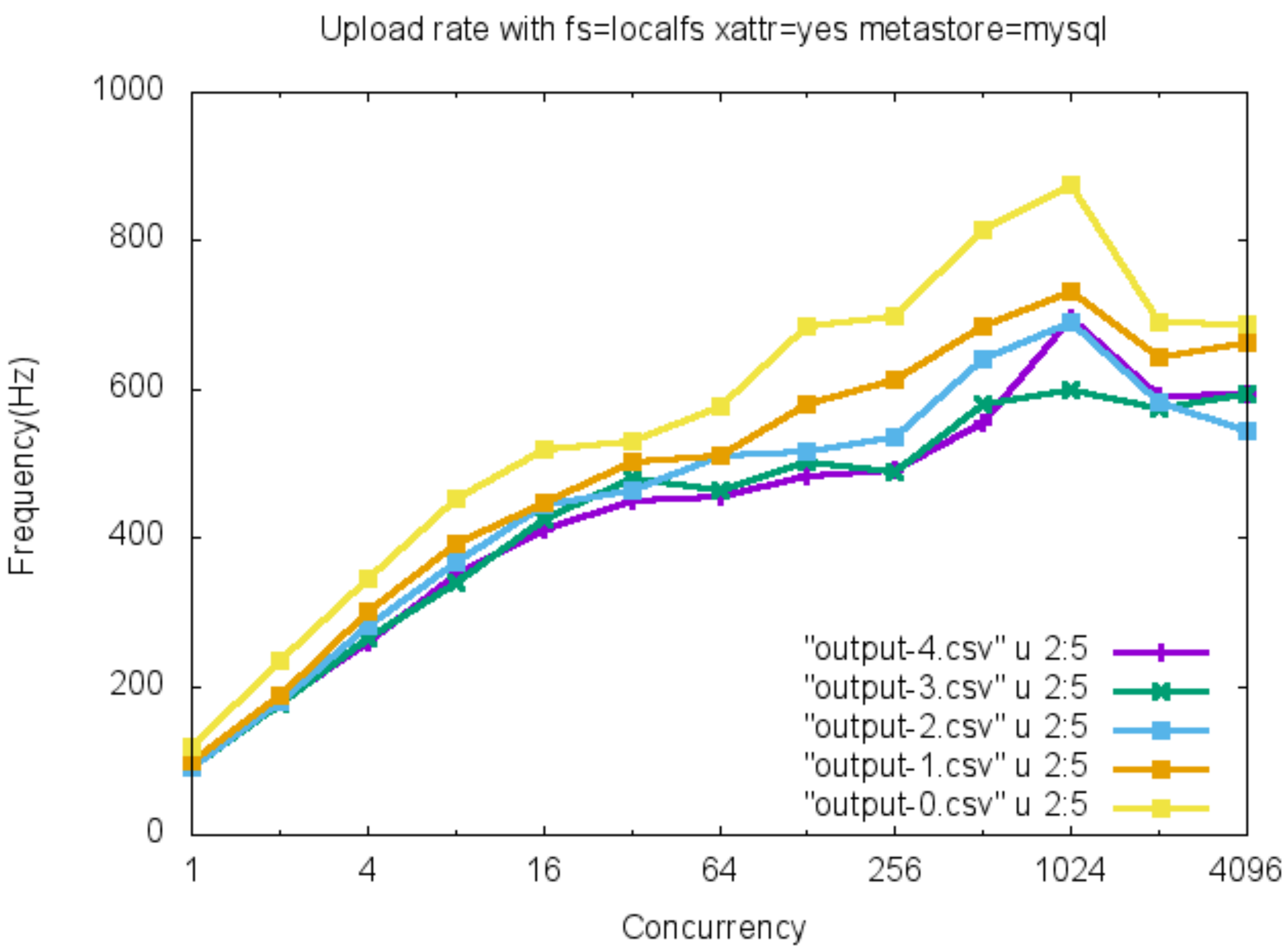
- Operation uses **HTTP/1.1**

- **Upload triggers metadata propagation.**

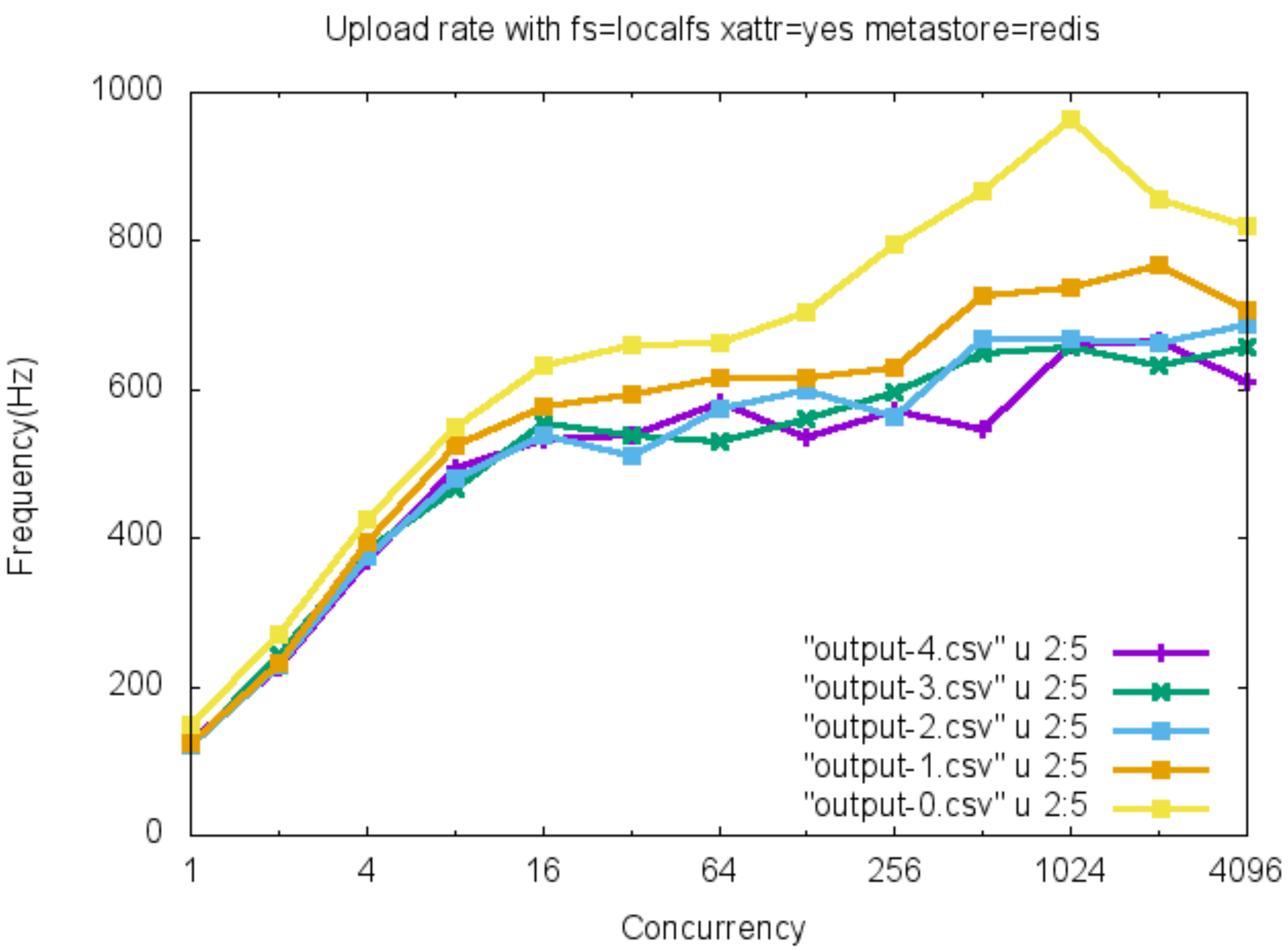
UPLOAD benchmark



UPLOAD benchmark

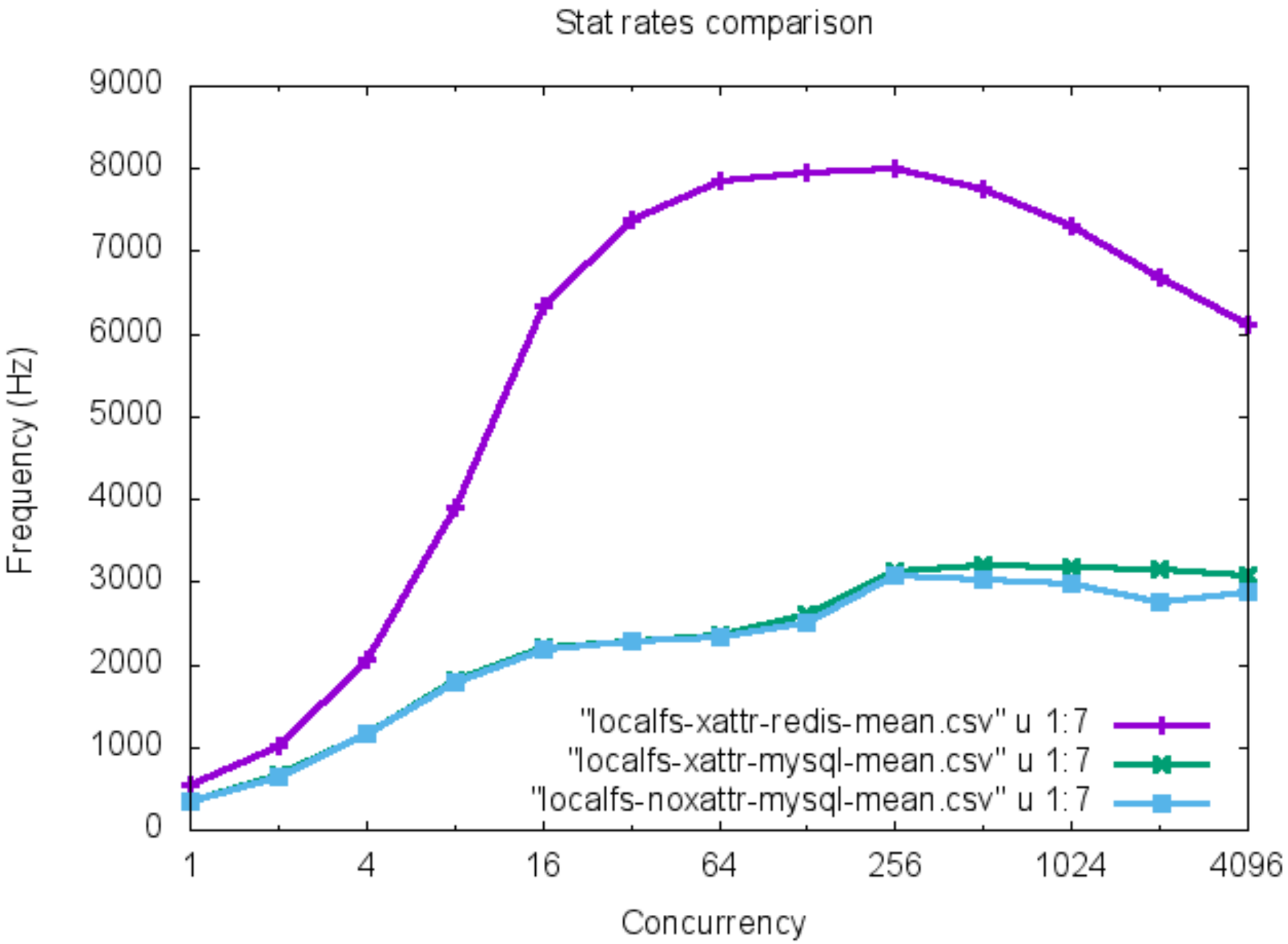


UPLOAD benchmark

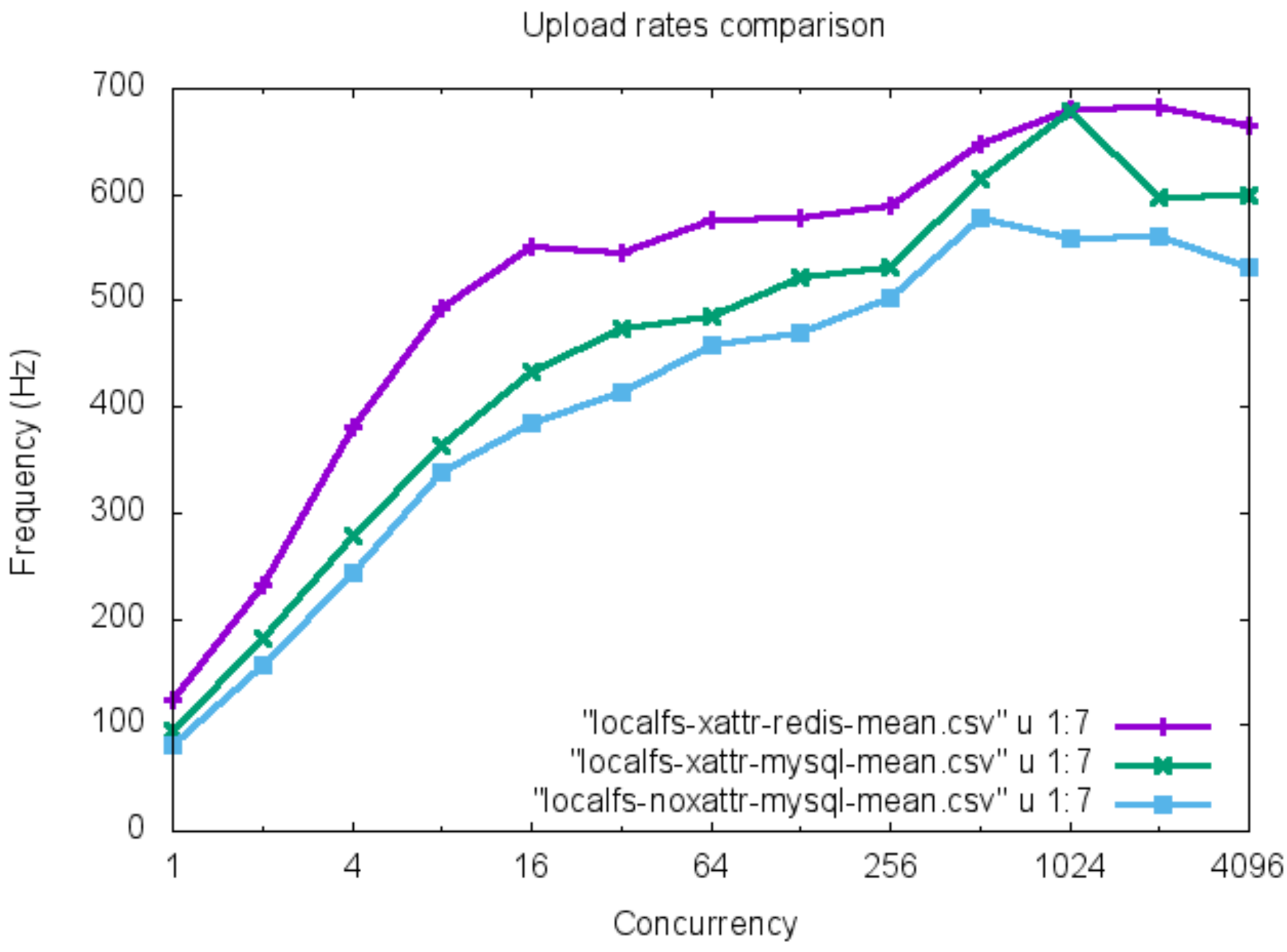


CONCLUSIONS

STAT COMPARISON



UPLOAD COMPARISON



What can we extract from these results ?

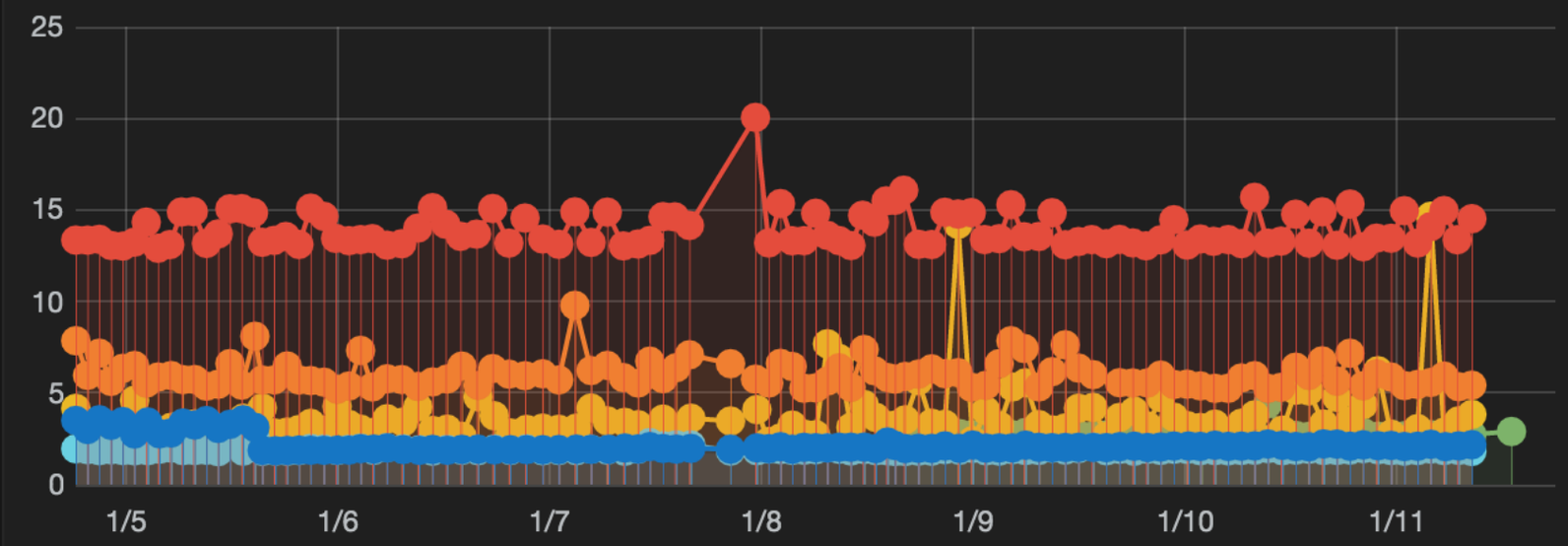
- Retrieval of file hierarchy from FS favours novel uses cases and access to existing data repositories.

- In-memory databases increase the performance and can scale to a high number or records (with a 70 bytes memory footprint per file, 64 GiB => 981714285714 files)

- Use of XATTRS makes the system more consistent

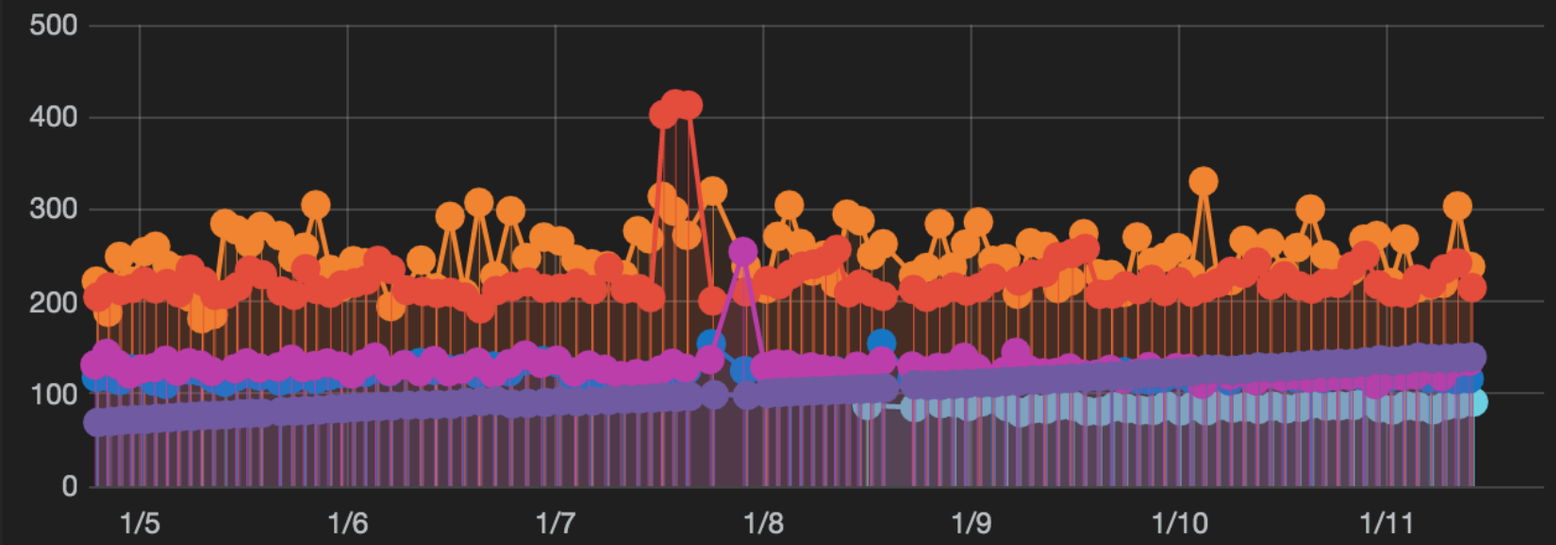
WE THINK WE ARE ON THE RIGHT WAY

SYNC TIME - 1 file - 1B



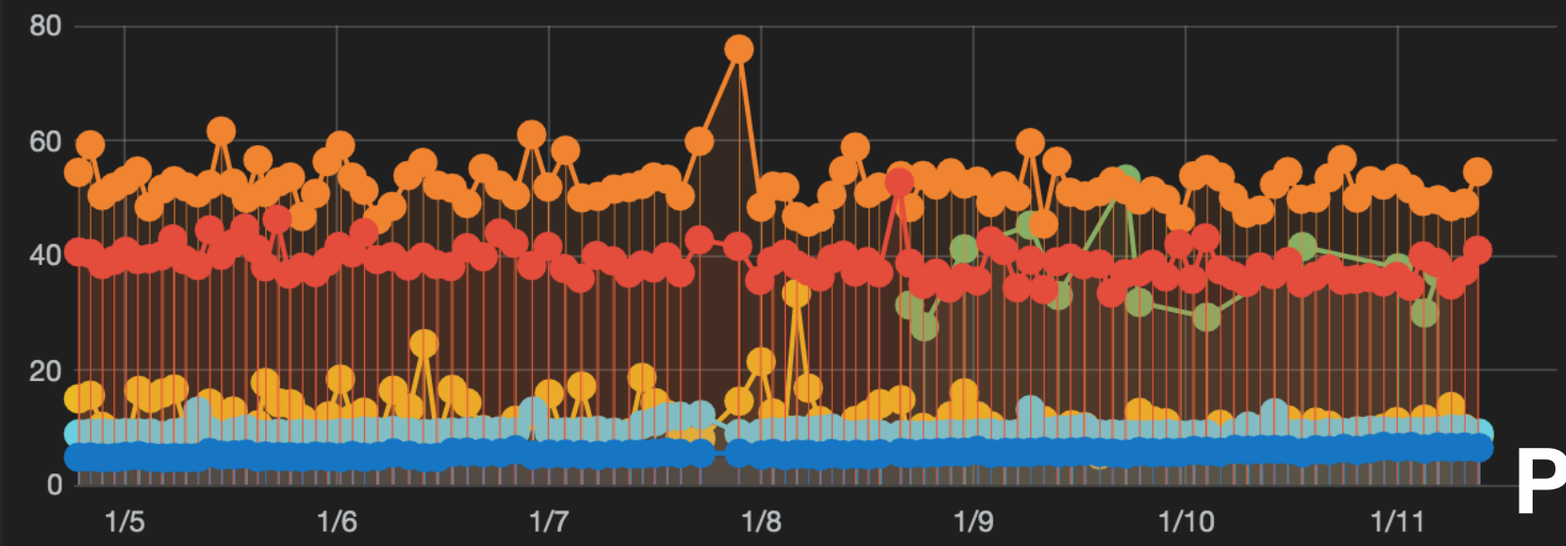
	avg
syncperf-total-syn {server_name: 193.147.87.143:57024}	2.78
syncperf-total-syn {server_name: cernbox.cern.ch}	3.73
syncperf-total-syn {server_name: data.deic.dk}	1.91
syncperf-total-syn {server_name: dropbox}	6.02
syncperf-total-syn {server_name: https://seacloud.cc/}	13.94
syncperf-total-syn {server_name: test.data.deic.dk}	2.21

SYNC TIME - 1000 files - 100kB



	avg
syncperf-total-syn {server_name: 193.147.87.143:57004}	106.14
syncperf-total-syn {server_name: 193.147.87.143:57014}	112.48
syncperf-total-syn {server_name: 193.147.87.143:57024}	86.24
syncperf-total-syn {server_name: cernbox.cern.ch}	250.00
syncperf-total-syn {server_name: data.deic.dk}	226.00
syncperf-total-syn {server_name: dropbox}	121.54
syncperf-total-syn {server_name: https://seacloud.cc/}	128.25
syncperf-total-syn {server_name: test.data.deic.dk}	103.76

SYNC TIME - 10 files - 10MB



	avg
syncperf-total-syn {server_name: 193.147.87.143:57024}	36.55
syncperf-total-syn {server_name: cernbox.cern.ch}	11.41
syncperf-total-syn {server_name: data.deic.dk}	9.37
syncperf-total-syn {server_name: dropbox}	52.31
syncperf-total-syn {server_name: https://seacloud.cc/}	38.72
syncperf-total-syn {server_name: test.data.deic.dk}	5.47

Piotr’s Analytics System

What comes next ?

What comes next ?

- **Improve performance** (more parallelisation)
- Run **more benchmarks**: upload with checksums, overwrites, remove, move, fuzz
- Perform benchmark on **cluster** (ClawIO design scales out)
- Implement more backends: **EOS, S3/SWIFT/RADOSGW**
  (plug your backend, suggestions are welcome)
- Implement **sharing** and run benchmarks on shared folders
- Test other **Sync Protocols**: SeaFile, StorageSync ?

Thank you

¿Q&A?

Acknowledgements

- Centro de Investigación, Transferencia e Innovación (CITI)

- PhD Jakub T. Mościcki @CERN

- Piotr Mrowczynski @Technical University of Denmark

- LIA2, University of Vigo