AeroFTP Community Benchmark first round (v3.7.3 / v3.7.4)

Community Benchmark, first round (v3.7.3 / v3.7.4)

> **Date**: 2026-05-07 / 2026-05-08 **CLI**: `aeroftp-cli 3.7.3` for the main sweep, `aeroftp-cli 3.7.4` for the post-fix verification round **Issue**: axpdev-lab/aeroftp#177 **Schema**: report v1 (see `docs/dev/roadmap/APPENDIX-BENCHMARK/01_JSON-Schema-v1.md`) **Dataset bundle**: gitignored at `/home/axpdev/aeroftp-bench-reports/2026-05-07/dataset/` (32 sanitized JSON reports + summaries + rankings)

This document is the maintainer-side reference run for the AeroFTP Community Benchmark initiative announced in issue #177. It is a single-host, single-residential-uplink sample and is published as a baseline, not as a population-level benchmark. Selection bias is explicit and called out in section 9.

---

1. Why this round exists

When AeroFTP claims that one protocol is faster than another on a given provider, that claim should be defensible. A single fiber line in a single timezone is not a credible base for a public protocol comparison page. Issue #174 invites the community to run the same matrix against their own profile and submit a sanitized JSON report.

Before asking other people to do that, the maintainer ran the matrix on every profile saved on the development host. The output of that exercise is this dataset and the bug fixes it surfaced.

---

2. Matrix

The main sweep used the new `benchmark custom` subcommand introduced in v3.7.3:

```
aeroftp-cli --profile "<name>" benchmark custom \
  --sizes 1M,10M,100M,1G --runs 3 \
  --consent-publish --report <out>.json
```

The verification round (post-v3.7.4 fixes for Filen, Yandex, Drime) used a smaller, faster matrix capped at 100 MiB:

```
aeroftp-cli --profile "<name>" benchmark custom \
  --sizes 1M,10M,100M --runs 2 \
  --consent-publish --report <out>.json
```

Each `(size, run)` tuple exercises five operations: **upload, download, list, stat, delete**. Numbers are reported as p50/p95/min/max/stddev per operation, not as arithmetic means. The CLI runs a sanitization pass before writing the JSON: if any path, hostname, account, bucket name, IP, MAC, token or fingerprint slips through, the report is rejected, not anonymized post-hoc.

---

3. Coverage

Class	Profiles	Notes
Full matrix (1M + 10M + 100M + 1G all OK)	AWS S3, Cloudflare R2, Alibaba OSS, S3 Storj, Tencent COS, S5 FileLu, Mega S3, Google S3, S3Drive, Drime, Internxt, Jotta, Koofr (native), FileLu, My Mega, Google Drive, Dropbox, OneDrive, pCloud, Azure Blob, Koofr WebDAV, FeliCloud, aeroftp.app FTPS	27 profiles
Partial (provider quota / size cap)	jianguoyun (CN, 1G refused), WebDAV DriveHQ (free quota), MyBox (Box 250 MB cap on the free tier), Internxt (10 GB quota saturated), MyZoho (1G blocked, 100M OK)	5 profiles
Hard-failed in v3.7.3, fixed in v3.7.4	Filen Dev (chunked AES-GCM), Yandex (transient `upload-target` race), Drime (`list()` mutated `current_path`)	3 profiles, re-run with v3.7.4
Out of scope this round	idrive S3 (cold-storage timeout), InfiniCloud jp (1G stuck at upload-target), kDrive, SeaFile WebDAV (no operations on root, need a sub-path matrix), Lumo NAS (powered off), Wasabi / Quotaless (access expired), 8 GitHub-as-storage profiles, 7 Aruba FTP duplicates, 3 media CDNs (ImageKit, Uploadcare, Cloudinary)	covered by separate handoffs or future rounds

`profiles-skipped.txt` documents the rationale for every profile that did not enter the deep manifest.

---

4. Bug fixes shipped because of this sweep

The sweep itself acted as a stress test of the rest of the codebase. Every defect surfaced was fixed before publishing the dataset. The fixes are split between the v3.7.3 patch queue (commits `253f2cc2` + `8e0f0b8f`) and the v3.7.4 release (`22a4bd8f`, `d16a63cc`, `cb1e80b6`).

Fix	Provider	Commit	Verified by
`benchmark_sanitize` substitutes PII before assertion	all	`8e0f0b8f`	FeliCloud, Azure Blob (reports written instead of rejected)
`SigpipeIgnoreGuard` wraps `cmd_benchmark`	all	`8e0f0b8f`	Azure Blob, S3 Backblaze, Yandex (no more rc=141)
OneDrive nested `mkdir` splits relative path on `/`	OneDrive	`253f2cc2`	OneDrive (full matrix in 178s)
`Drime::list()` no longer mutates `current_path`	Drime	`253f2cc2`	Drime (full matrix in 65s post-fix)
HTTP `read_timeout` 300s → 1800s on all 24 providers	all HTTP-based	`253f2cc2`	Koofr WebDAV (1G upload in 1010s instead of dying at 5min)
Chunked AES-GCM upload (1 MiB / `index=N`)	Filen native	`22a4bd8f`	Filen Dev (10M+ no longer hits 413)
Per-chunk retry on egest body decode failures	Filen native	`d16a63cc`	Filen Dev (transient `decode body` no longer fatal)
Retry upload PUT with fresh `upload-target` on transient failures	Yandex Disk	`cb1e80b6`	Yandex (100M no longer single-PUT race)

The sanitizer fix is structural: before, a hostname leaking past the per-provider redactor would refuse the report. After, the sanitizer substitutes the offending substring with a coarse hint and the report is written. We prefer one extra coarse field over zero data.

---

5. Top performers

The numbers below are from the main 1 GB matrix where available and the 100 MiB matrix otherwise. They are p50 across 3 runs, not means. Full per-payload tables live in `dataset/rankings/rankings-custom1g.md`.

Upload (p50 Mbps, 1 GiB payload)

#	Profile	p50	p95
1	pCloud	281.8	285.3
2	Jotta	271.2	276.6
3	Dropbox	240.5	241.3
4	Alibaba OSS	209.3	238.5
5	aeroftp.app (FTPS)	201.0	212.3
6	S3 Storj	188.1	189.9
7	Cloudflare R2	185.0	190.1
8	FileLu	183.8	187.9
9	Tencent COS	178.0	198.3
10	Google S3	167.8	184.2

Download (p50 Mbps, 1 GiB payload)

#	Profile	p50	p95
1	Google S3	585.1	616.3
2	Mega S3	584.9	588.4
3	Tencent COS	569.3	596.0
4	Alibaba OSS	556.5	561.2
5	Jotta	471.1	475.2
6	FileLu	412.8	423.5
7	Google Drive	304.5	336.2
8	Cloudflare R2	286.6	291.4
9	My Mega	229.8	235.4
10	Koofr WebDAV	210.0	213.9

Two non-obvious takeaways:

1. **Asymmetry is real**. The same provider can rank top-3 on upload and bottom-3 on download (Azure Blob), or vice-versa (Google S3). A single-direction benchmark would mislead.
2. **The S3 protocol is not universally fastest**. Native APIs (pCloud, Jotta, Dropbox) win the 1 GiB upload bracket. The S3-as-backend assumption breaks here.

---

6. Verification round (post-v3.7.4)

Three profiles failed in the main v3.7.3 sweep with errors that were *AeroFTP bugs*, not provider limits. The v3.7.4 release plus one in-queue commit (`cb1e80b6`) fixed them. The verify round below was run on a v3.7.4 build that includes that pending commit, with the 100 MiB cap matrix (`custom --sizes 1M,10M,100M --runs 2`) to confirm the fixes hold without re-burning bandwidth on 1 GiB payloads.

Profile	v3.7.3 status	v3.7.4 result (verify, 100 MiB cap)	Fix commit
Filen Dev	failed at 10 MB upload (`413 Payload Too Large`)	**OK**: 21 runs / 444 MB / 195 s, 0 transient + 0 fatal errors. Upload p50: 4.77 / 24.29 / 30.83 Mbps at 1M/10M/100M. Download p50: 34.22 / 28.99 / 29.22 Mbps	`22a4bd8f` , `d16a63cc`
Drime	mid-sweep dir corruption: list() mutated `current_path` , every operation after upload landed on a non-existent path	**OK**: 21 runs / 444 MB / 87 s, 0 transient + 0 fatal errors. Upload p50: 4.01 / 19.26 / 64.95 Mbps at 1M/10M/100M. Download p50: 6.38 / 53.47 / 259.88 Mbps	`253f2cc2`
Yandex	failed at 100 MB upload (no retry on transient `upload-target`)	**OK** at 1 MB (7 runs, 36 s, 0 errors). Upload p50: 0.87 Mbps, download p50: 5.92 Mbps. The Yandex free-tier server caps upload throughput around 1 Mbps server-side, so 10 MB and 100 MB single-shot PUTs are closed mid-stream regardless of retry logic. The `cb1e80b6` fix is therefore validated where it was meant to apply (transient failures); larger payloads need chunked resumable upload with `Content-Range` , which stays open as a separate work item	`cb1e80b6`

Verify-round JSON reports land in `dataset/reports/<profile>_verify.json` . The maintainer's runner is `run-verify.sh` in the bundle root.

While re-running Drime, a smaller bug was surfaced and fixed in the same session: Drime's `mkdir()` returned a generic `ServerError` when the API answered `422 Unprocessable Entity` with `"Folder with same name already exists."`, so the `benchmark` command logged a cosmetic warning even though the directory existed and the run continued normally. The provider now maps that exact response to `ProviderError::AlreadyExists`, which the benchmark already treats as idempotent. Pending in queue for the next release.

---

7. Known open bugs not in this round

Bug	Provider	Status	Tracking
`Y3` (emerged after Y2 close)	Yandex Disk	OPEN	`docs/dev/email_providers/yandex_workfolder/HANDOFF.md`
Benchmark assumes overwrite-on-PUT	4shared	OPEN	CLI should `delete` between runs for strict providers
1G timeout cap for slow storage	idrive S3, InfiniCloud jp	OPEN	bench should accept `--per-profile-timeout` flag
No-root path matrix	kDrive, SeaFile WebDAV	OPEN	both providers refuse operations on `/`, need a sub-path benchmark variant
S5 FileLu native `delete` returns 500	FileLu native	OPEN	provider-side intermittent, not reproducible on demand

---

8. How to reproduce

```
# Same CLI binary or rebuild from main post-v3.7.4
CLI=/var/www/html/FTP_CLIENT_GUI/src-tauri/target/release/aeroftp-cli

# Single profile
"$CLI" --profile "AWS" benchmark custom \
  --sizes 1M,10M,100M,1G --runs 3 \
  --consent-publish --report ./AWS.json

# Full sweep (uses the manifest in this dataset)
cd /home/axpdev/aeroftp-bench-reports/2026-05-07
./run-benchmark.sh custom1g profiles-deep.txt
```

The `--consent-publish` flag wraps the JSON in `BEGIN AEROFTP BENCHMARK REPORT` / `END AEROFTP BENCHMARK REPORT` markers and runs the sanitization sweep. If anything that would identify the host or the account makes it past the per-provider redactor, the command refuses to write the report. You will not produce a poisoned report by accident.

---

9. Selection bias and disclaimers

This dataset was produced on:

- **Single host** (Linux x86_64, kernel 6.8, Ubuntu 24.04 LTS)
- **Single uplink** (residential WE fiber, asymmetric ~4 MB/s up / ~3.5 MB/s down nominal, although the rankings show several runs well above that, indicating provider-side compression or per-stream burst boost)
- **Single timezone** (UTC+02, all runs in one ~36 h window)
- **Single AeroFTP version** (v3.7.3 build for the main sweep, v3.7.4 build for the verify round)

It is a baseline for *what one developer measures from one location with one binary*. The community benchmark page on `docs.aeroftp.app` will only draw protocol-level conclusions after aggregating community submissions across at least three regions and two connection types. Any conclusion drawn from this dataset alone is provisional.

---

10. Files in this subdirectory

- `HANDOFF.md` : handoff between work sessions during this round
- `REPORT.md` : this file
- `run-benchmark.sh` (in the gitignored bundle): the runner
- `dataset/` (in `/home/axpdev/aeroftp-bench-reports/2026-05-07/`): 32 sanitized JSON reports, 4 CSV summaries, rankings, manifests

The dataset bundle is **not** committed to git: it is the deliverable that gets attached to issue #177 by the maintainer, referenced by URL from the community benchmark page once aggregation across submissions is done.

Last updated: 2026-05-08