SERIES 0-DAY NO CVE HIGH PUBLIC

Stirling-PDF 3.2.0, processing-folder implied storage root IDOR

Stirling-PDF 3.2.0 FolderAccessGuard implies the whole ./storage tree. ROLE_USER points a processing folder at storage/<victimId>, lists that owner's PDFs, and overwrites them. Direct storage GET stays 403. No CVE yet.

03 SERIES Stirling-PDF v3.2.0 leftovers

00 Stirling-PDF 3.2.0, H2 CREATE OR REPLACE ALIAS leftover of GHSA-mrr8

00 The pin

00 Stirling-PDF 3.2.0, processing-folder implied storage root IDOR

← PREVIOUS
Phase 0: The pin

NAME	SERIES
Stirling-PDF 3.2.0, processing-folder implied storage root IDOR	Stirling-PDF v3.2.0 leftovers
PHASE	TYPE
0	0-day analysis
CVE	CVE RISK
n/a	high
DISCLOSURE STATUS	VENDOR
public	Stirling-PDF

AFFECTED	PUBLISHED
Stirling-PDF 3.2.0 (a7e593728dd9aa81e75318a949789d3c2e9ee754), image stirlingtools/stirling-pdf:3.2.0-fat, login-on, storage.enabled=true.	**17 Oct 2026**
UPDATED	**TAGS**
17 Oct 2026	**0-day, stirling-pdf, idor, cwe-639, leftover, remote**

TL;DR

Lab files and proof-of-concept here.

01 **They scoped /api/v1/storage/files/{id}. They left the disk path next to it.**

I am @abraxas_null. The proof of concept is on GitHub: **abraxas/stirling-storage-folder-idor** (loopback client). The lab stack is `lab/`: docker-compose.yml, run.sh, poc.py. Authorized lab only. It talks to loopback.

This is leftover #2 of the Stirling-PDF v3.2.0 leftovers series. Sibling: stirling-h2-import-leftover (`CREATE OR REPLACE ALIAS` leftover of GHSA-mrr8). H2 is first. This post is the storage IDOR.

This is **Stirling-PDF v3.2.0** (`a7e593728dd9aa81e75318a949789d3c2e9ee754`), image `stirlingtools/stirling-pdf:3.2.0-fat`, login-on, `storage.enabled=true`. No CVE yet. CWE-639 / CWE-73. 7.1 (`CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:L/I:H/A:N`). **Reach:** Remote ROLE_USER. **Class:** IDOR.

`POST /api/v1/processing-folders` has no `@PreAuthorize` beyond a logged-in user. `FolderAccessGuard` implies the whole local storage `basePath` (`./storage`)

even when `policies.allowedFolderRoots` is empty. `LocalStorageProvider` writes `{ownerId}/{uuid}_{filename}`. A ROLE_USER can set `directory: "storage/<victimUserId>"`, list that owner's PDFs, and sweep them in place with `replace=true`.

Direct `GET /api/v1/storage/files/{id}` is patched (`findByIdAndOwner`). Lab GET of the victim file as the peer is 403. The leftover is the processing-folder implied root, not the storage CRUD API.

A ROLE_USER lists another user's stored PDF names and overwrites those blobs. The victim download returns the tampered file.

02 Why Stirling-PDF

3.x added local storage and processing folders. Storage CRUD learned `findByIdAndOwner`. Processing folders take a disk `directory`. `FolderAccessGuard.serverStorageRoots` returns `Path.of(basePath)` for the entire tree, not `basePath/{currentUserId}`. `storage/<victimId>` `startsWith` that implied root.

Official compose is login-off plus `DISABLE_ADDITIONAL_FEATURES=true`. That deploy 403s `/api/v1/storage` and omits this controller. Template login-on plus fat is the HIT deploy. `storage.enabled` defaults true.

Vendor intake is the GitHub Security Advisory tab or security@stirlingpdf.com. Unpaid. I am not opening a public issue on Stirling-Tools/Stirling-PDF.

03 Pinning v3.2.0

Latest GA. Tag v3.2.0. Peel is `a7e593728dd9aa81e75318a949789d3c2e9ee754`. Image `stirlingtools/stirling-pdf:3.2.0-fat`. Do not treat "I patched `/api/v1/storage/files/{id}`" as "I scoped every API that takes a disk path under `./storage`."

The lab binds **only** `127.0.0.1:18154`. Two users on the default team. Witness in a watermark stream.

04 The skip list is half the hunt

Direct storage IDOR is patched. Folder UUID CRUD is `findByIdAndOwner`. `LocalStorageProvider` rejects `..` / absolute keys after normalize. Config dir is a protected root. Ultra-lite omits the controller. Login-off 403s storage. Public GHSAs on this tree are conversion SSRF, H2 import strings, pipeline INTERNAL_API_USER, zip-slip. None of them is this processing-folder path.

Public n-day issue #7628 `.fodt` / `.fods` is still unpatched. Do not "discover" it.

05 Surfaces that returned nothing

Direct `GET /api/v1/storage/files/1` as `peeruser` is 403. Un-ranged victim download after overwrite 500s because advertised DB `sizeBytes` is stale. Range works. Byte copy into an attacker-readable `outputId` was not required for listing/overwrite. `GET /api/v1/user/users` listed `STIRLING-PDF-BACKEND-API-USER` userId=2. Extra IOC. I did not file it as its own leftover.

The temperature changed on `serverStorageRoots` returning the entire base.

06 The implied root is the whole argument

`FolderAccessGuard.serverStorageRoots`:

PLAIN TEXT Copy

```
return List.of(normalize(Path.of(basePath)));
```

Implied roots are always permitted, even with no configured `allowedFolderRoots`. `FolderInputSource.validate` calls `requirePermitted`. `storage/1` starts with `./storage`. `GET /{id}/files` does `Files.list` on that directory. Extensions pinned to `.pdf`. Sweep writes back with `replace=true`.

Lab: victim `admin` id 1 uploaded `victim-lab.pdf` (storage file id 1, 571 bytes). Attacker `peeruser` id 3 `ROLE_USER`. Direct storage GET 403. `POST /api/v1/processing-folders` `{directory:"storage/1"}` 202, folder id `8adc1fe4-0bfd-421f-b661-92ef059b7449`. `GET …/{id}/files` listed `5341de35-8729-49cf-92d0-c8284e05bfc5_victim-lab.pdf`. Sweep `/api/v1/security/add-watermark` with the witness. Folder file went `done` at 3450 bytes. Victim Range download 206 / 3450 contains `STIRLING-STORAGE-FOLDER-IDOR-WITNESS` after inflate of Identity-H FlateDecode streams.

07 The lab

Source of truth is `lab/`. Image `stirlingtools/stirling-pdf:3.2.0-fat`. Port **18154**. Bind it to loopback.

PLAIN TEXT Copy

```
git clone https://github.com/abraxas/stirling-storage-folder-idor
cd stirling-storage-folder-idor
python3 stirling-storage-folder-idor-Abraxas-Labs.py
```

or `cd lab && ./run.sh`. The loopback client is `lab/run.sh` with a banner on it. Target **only** `http://127.0.0.1:18154`. Do not publish the port off loopback.

Last lab run, trimmed:

PLAIN TEXT Copy

```
IOC direct-storage-get http=403
IOC processing-folders-save directory=storage/1 http=202
```

```
IOC listed-peer ... victim-lab.pdf
IOC victim-download attempt=1 http=206 bytes=3450 witness=1
SUCCESS stirling-storage-folder-idor direct-storage-403=yes listed-peer-pdf=yes c
```

Witness string: `STIRLING-STORAGE-FOLDER-IDOR-WITNESS`. It is watermark text. The oracle inflates PDF streams. License grandfathered 5 users.

08 Wrong turns, in the order I made them

Treating direct storage IDOR as still open. 403. Last line requires `direct-storage-403=yes`.

Treating login-off official compose as the HIT deploy. Storage 403, controller absent. Fat plus login-on is the deploy.

Claiming attacker-readable byte copy without an `outputId`. Listing and in-place rewrite do not need it. I kept byte copy as inferred.

Un-ranged download as the oracle. 500 after overwrite (stale `sizeBytes`). Range 0-1048575 is 206 with the rewritten blob.

Filing INTERNAL_API_USER on `GET /api/v1/user/users` as a High leftover. Extra IOC. Username enum, not stored-PDF IDOR.

09 What an attacker can do

Any logged-in human `ROLE_USER` on login-on proprietary/fat with local storage enabled.

Privilege table (exhaustive). Same bar as Signal-Server Dropwizard admin.

Recon (lab-proven)

METHOD + PATH	
ROLE_USER `GET /api/v1/user/users`	
`POST /api/v1/processing-folders` `{directory:"storage/1"}` then `GET /api/v1/processing-`	

Integrity (lab-proven)

METHOD + PATH	
`POST /api/v1/processing-folders/{id}/sweep` watermark step, then victim Range download	

Confidentiality (inferred)

METHOD + PATH	GRANTS	PROVEN
same sweep plus a team `outputId` / outbound step the attacker can read	Copy of victim PDF bytes	Inferre Not requir

Adjacent surface that stays closed

METHOD + PATH	WHY IT IS CLOSED
`GET /api/v1/storage/files/{id}` of a non-shared file	403. Lab-proven on file id 1
unauth `/api/v1/storage/**`	403 when login is off
folder UUID CRUD across owners	`findByIdAndOwner`
`LocalStorageProvider` `..` / absolute key	`Path.startsWith(basePath)` after normalize
ultra-lite image	proprietary controller absent

No password change. No admin mint. No shell. Confidentiality of stored PDF names plus integrity of those blobs: High. That is why it is in bar.

10 What this is not

It is not "storage CRUD never shipped an owner check." Direct file GET is 403. This processing-folder directory is not. It is not unauthenticated. It is not a reverse shell. It is not lab-proven attacker-readable byte copy.

11 The fix

Scope the implied storage root to `basePath/{currentUserId}` for disk `directory` on multi-user servers. Refuse `storage/<otherId>` the way `findByIdAndOwner` already refuses foreign file ids. Recompute advertised `sizeBytes` after an in-place rewrite.

Re-run the loopback client against a patched build: `directory=storage/1` as `peeruser` must 403, and listed-peer must stay empty.

Until then, do not treat "I patched `/api/v1/storage/files/{id}`" as "I scoped every API that takes a disk path under `./storage`."

I am not going to print a walk of `storage/1` aimed at a live host. The implied root is the useful part. If you own the box, run the proof of concept against loopback.

Same product, other unpublished lab on this tag: H2 CREATE OR REPLACE ALIAS. Series hub: Stirling-PDF v3.2.0 leftovers.

12 References

- Proof of concept: abraxas/stirling-storage-folder-idor · stirling-storage-folder-idor-Abraxas-Labs.py
- Lab: `lab/` · docker-compose.yml · run.sh · poc.py
- Series: Stirling-PDF v3.2.0 leftovers · stirling-h2-import-leftover
- @abraxas_null · github.com/abraxas · abraxaslabs.tech · abraxas.null@proton.me
- CWE-639 · CWE-73
- Tree: Stirling-PDF v3.2.0 (`a7e593728dd9aa81e75318a949789d3c2e9ee754`) · `FolderAccessGuard.java` · `ProcessingFolderController.java`

- Product: Stirling-PDF

03 SERIES Stirling-PDF v3.2.0 leftovers

00 Stirling-PDF 3.2.0, H2 CREATE OR REPLACE ALIAS leftover of GHSA-mrr8

00 The pin

00 Stirling-PDF 3.2.0, processing-folder implied storage root IDOR

← PREVIOUS

Phase 0: The pin

A X Abraxas Labs

Independent security research by abraxas.

Analysis of 0-days, n-days, malware, and the

leftovers that still matter.

GitHub @abraxas_null abraxas.null@proton.me abraxaslabs.tech

Independent research. Private analysis by arrangement. No ads. abraxas.null@proton.me · For defensive awareness and authorized testing only.

© 2026 Abraxas Labs