Flare Zero day Research

- Private File IDOR via raw/direct endpoints
- Password?Protected Thumbnail Bypass

Private File IDOR via raw/direct endpoints

GHSA : https://github.com/FlintSH/Flare/security/advisories/GHSA-gwqr-xf5c-5569

CVE : CVE-2026-30231

Summary

The raw and direct file routes only block unauthenticated users from accessing private files. Any authenticated, non?owner user who knows the file URL can retrieve the content, which is inconsistent with stricter checks used by other endpoints

ezgif-80eb9d3b19d68298.gif

Evidence (Code References)

- raw route predicate: raw/route.ts:L92-L97
- direct route predicate: direct/route.ts:L35-L40
- download route correct check: download/route.ts:L36-L45
- thumbnail route correct check: thumbnail/route.ts:L32-L45

Impact

- Confidential private files can be accessed by any authenticated user if the URL is known.
- Violates access control consistency across file endpoints.

Expected vs Actual

- Expected: Private files should be accessible only to the owner or admin.
- Actual: Private files are accessible to any authenticated user on raw/direct routes.

Minimal Logic Reproduction (non?network)

This demonstrates the misclassification using the same predicates.

```
type Session = { user?: { id: string; role?: 'ADMIN' | 'USER' } } | null

function rawIsPrivate(visibility: 'PUBLIC' | 'PRIVATE', session: Session) {
  return visibility === 'PRIVATE' && !session?.user
}

function directIsPrivate(visibility: 'PUBLIC' | 'PRIVATE', session: Session) {
  return visibility === 'PRIVATE' && !session?.user
}

// Setup: private file owned by A; authenticated B session
const fileVisibility = 'PRIVATE' as const
const sessionB: Session = { user: { id: 'user-B', role: 'USER' } }

// Current behavior in raw/direct:
rawIsPrivate(fileVisibility, sessionB)     // false → grants access
directIsPrivate(fileVisibility, sessionB)  // false → grants access

// Expected: deny unless owner or admin
function expectedIsPrivate(
  visibility: 'PUBLIC' | 'PRIVATE',
  session: Session,
  ownerId: string
) {
  const isOwner = session?.user?.id === ownerId
  const isAdmin = session?.user?.role === 'ADMIN'
  return visibility === 'PRIVATE' && !isOwner && !isAdmin
}
```

Affected Components

- raw file endpoint: raw/route.ts
- direct file endpoint: direct/route.ts

Remediation

Align raw/direct with download/thumbnail:

```
const isOwner = session?.user?.id === file.userId
const isAdmin = session?.user?.role === 'ADMIN'
const isPrivate = file.visibility === 'PRIVATE' && !isOwner && !isAdmin

if (isPrivate) {
  return new Response(null, { status: 404 }) // or 403
}
```

Consider centralizing access checks in a shared utility to ensure consistency across endpoints.

Verification Checklist

- Private file as Owner A:
  - Unauthenticated user ? denied (404/401)
  - Authenticated non?owner User B ? denied
  - Admin ? allowed
  - Owner ? allowed
- Behavior matches download/thumbnail routes.
- Automated tests added for owner/admin/non?owner/unauthenticated across raw/direct.

Password?Protected Thumbnail Bypass

GHSA : https://github.com/FlintSH/Flare/security/advisories/GHSA-3x7v-x3r6-mjh7

CVE : CVE-2026-30230

Summary

The thumbnail endpoint does not validate the password for password?protected files. It checks ownership/admin for private files but skips password verification, allowing thumbnail access without the password.

Affected Component

- Thumbnail endpoint: thumbnail/route.ts

Evidence (Code References)

- File password is fetched but never checked: thumbnail/route.ts:L32-L49
- Password checks exist in other endpoints:
- Download: download/route.ts:L50-L67
- Raw: raw/route.ts:L99-L107

Video POC

Your browser does not support the video tag.

Impact

- Visual content of password?protected files can be previewed through thumbnails without the password.
- Information disclosure of sensitive images despite password protection.

Expected vs Actual

- Expected: Password?protected files require a valid password for any content access, including thumbnails.
- Actual: Thumbnail content is served without password verification.

Reproduction Checklist

- Create User A and upload an image with a password.
- Note the file ID.
- Log in as User B (non?owner, non?admin).
- Request the thumbnail for User A’s file without providing the password.
- Expected: access denied.
- Actual: thumbnail returned.

Consider aligning thumbnail checks with the download/raw endpoints for consistent behavior.

Verification Checklist

- Create a password?protected image file.
- Access thumbnail as:
  - Unauthenticated user ? denied
  - Authenticated non?owner ? denied unless password provided
  - Owner/admin ? allowed
- Confirm behavior matches download/raw endpoints.