

 immich-app / immich Public[Code](#) [Issues](#) 523 [Pull requests](#) 183 [Discussions](#) [Actions](#) [Projects](#)

# Insecure transmission of shared link password

Moderate jrasrn91 published GHSA-78x4-6x83-jx75 2 weeks ago

## Package

 immich-server [\(Docker\)](#)

## Affected versions

&lt; v2.5.2

## Patched versions

&gt;= 2.6.0

## Description

### Summary

The Immich application is vulnerable to credential disclosure when a user authenticates to a shared album. During the authentication process, the application transmits the album password within the URL query parameters in a GET request to `/api/shared-links/me`. This exposes the password in browser history, proxy and server logs, and referrer

headers, allowing unintended disclosure of authentication credentials. The impact of this vulnerability is the potential compromise of shared album access and unauthorized exposure of sensitive user data.

### Details

The application transmits and exposes sensitive authentication credentials and user data during the shared album authentication process. Specifically, the `/api/shared-links/me` endpoint accepts the album password as a query parameter within the request URL and subsequently returns it in the HTTP response body alongside sensitive information about the album owner.

Example:

```
GET /api/shared-links/me?password=admin@44&slug=private
```



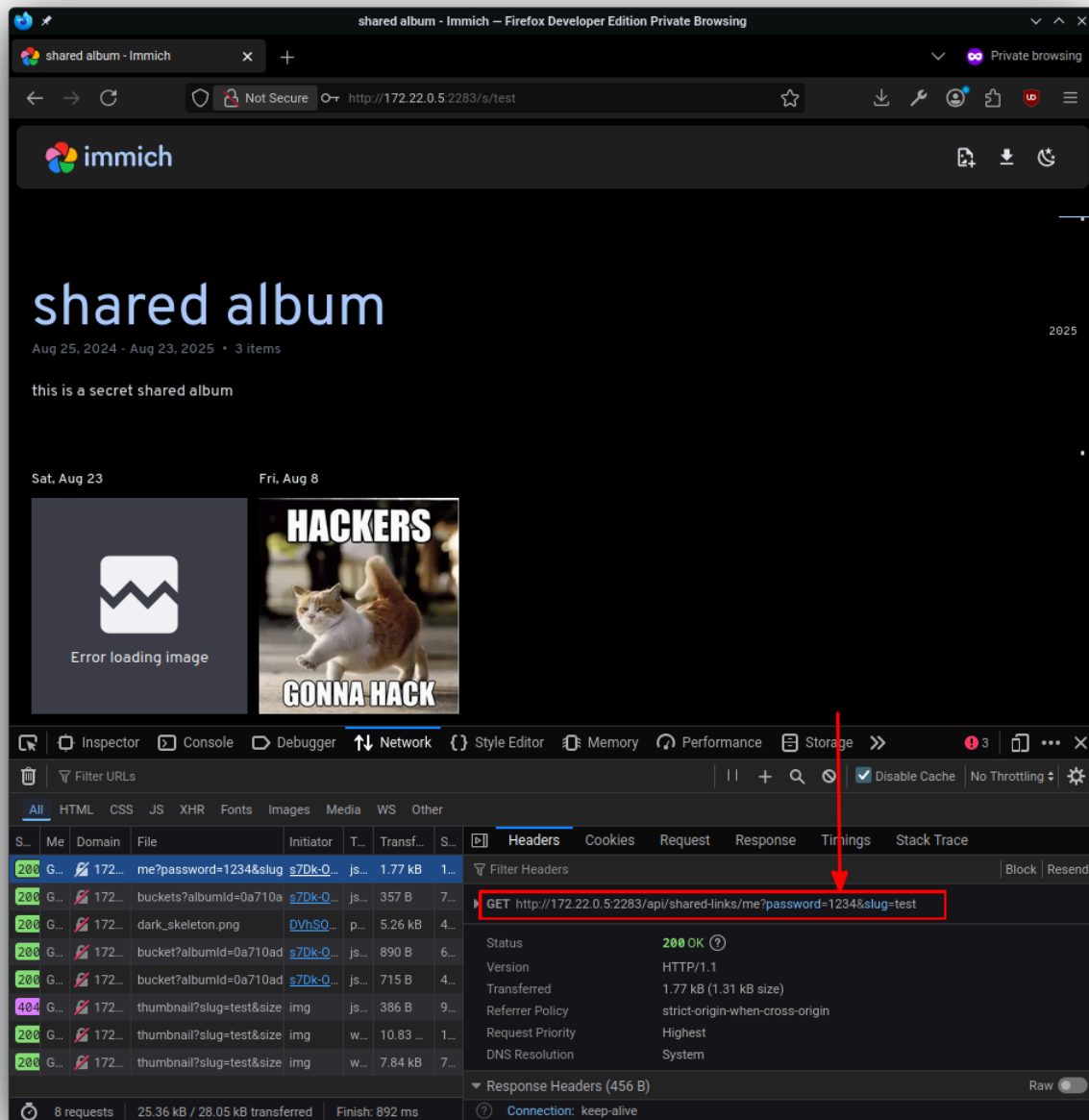
This behavior results in multiple layers of data exposure. The inclusion of the password in the request URI causes it to appear in browser history, reverse proxy logs, and referrer headers, while the server response further discloses sensitive details including user identifiers, email address, and more. As a result, an unauthenticated attacker with network access, access to logs, or control of intermediate systems could obtain the album password which leads to gaining access to album media content, and related personal data, compromising the confidentiality of private content and user information.

## PoC

### Steps to reproduce

1. Login as an admin to the Immich application.
2. Click on Sharing > Create album.
3. Fill in the album details including a title, description, and select an already uploaded image.
4. Click on the “Share” button in the top right corner.
5. Click on “Create Link”.
6. Enter the details for the custom URL and password.
7. Click on “Create Link”.
8. This will show the QR code and the URL generated to access the album.
9. Visit the link in an incognito browser window (without any previous session cookies.)
10. Open the network tab in the developer mode in the browser.
11. Visit the link and enter the password to authenticate to the Album.
12. Observe how the request transmits the password as a query parameter in the GET request URL.

### Proof of Concept (Image)



The screenshot shows a web browser displaying a "shared album" page. The page title is "shared album" and it includes the text "this is a secret shared album". Below the text, there are two image thumbnails: one with a "Error loading image" message and another with a cat and the text "HACKERS GONNA HACK". The browser's developer tools are open, showing the Network tab. A request to `http://172.22.0.5:2283/api/shared-links/me?password=1234&slug=test` is highlighted with a red box, and a red arrow points to it from the right.

## Recommendations

The `/api/shared-links/me` endpoint should be refactored to use the POST method and handle authentication credentials within the HTTP request body transmitted over HTTPS. The application must not include the password field in either the request URL or the API response. Upon successful authentication, the server should return only the necessary access token required for session validation. All sensitive fields, including password, userId, email, and owner information, should be excluded from API responses to prevent unnecessary exposure of personal or security-relevant data

## Severity

Moderate 6.3 / 10

**CVSS v4 base metrics**

**Exploitability Metrics**

Attack Vector	Network
Attack Complexity	Low
Attack Requirements	Present
Privileges Required	None
User interaction	None

**Vulnerable System Impact Metrics**

Confidentiality	Low
Integrity	None
Availability	None

**Subsequent System Impact Metrics**

Confidentiality	None
Integrity	None
Availability	None

[Learn more about base metrics](#)

CVSS:4.0/AV:N/AC:L/AT:P/PR:N/UI:N/VC:L/VI:N/VA:N/SC:N/SI:N/SA:N

**CVE ID**

CVE-2026-25118

**Weaknesses**

▶ CWE-598

**Credits**



rvizx

Reporter