Waqar-Arain / **Stored_HTML_Injection_due_to_SVG_File_Upload.md** Secret

Last active 3 weeks ago

Code Revisions 4

Stored_HTML_Injection_due_to_SVG_File_Upload.md

Field	Value
Product	66biolinks by AltumCode
Affected Version	61.0.1
CVE ID	CVE-2025-66939
Description	Stored HTML injection in 66biolinks v61.0.1 by AltumCode allows users to upload SVG files as favicons for their Biolink pages. While server-side sanitization removes some HTML elements, it does not filter or neutralize and tags within the SVG. This allows an attacker to upload a crafted SVG containing arbitrary HTML that will be rendered when other users visit the page, enabling stored HTML injection, external resource loading for tracking, phishing via links, and potential referrer leakage.
Vulnerability Type	CWE-79: Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')

Description

The application allows users to upload SVG files as favicons for their “Biolink pages”. Server-side sanitization removes several HTML elements but fails to filter or neutralize (anchor) and tags embedded inside the SVG. As a result, an attacker can upload an SVG containing arbitrary HTML elements that will be rendered in other users’ visiting favicon links.

Although JavaScript execution does not occur due to sanitization rules, the presence of unfiltered HTML tags still allows:

- Stored HTML injection
- External resource loading through , allowing user tracking and external request generation
- Phishing vectors through links, allowing attacker-controlled URLs to be displayed in a trusted UI context
- Referrer leakage, exposing internal URLs or user identifiers to third-party servers
- Potential escalation if sanitization bypasses or future browser behavior changes

Impact

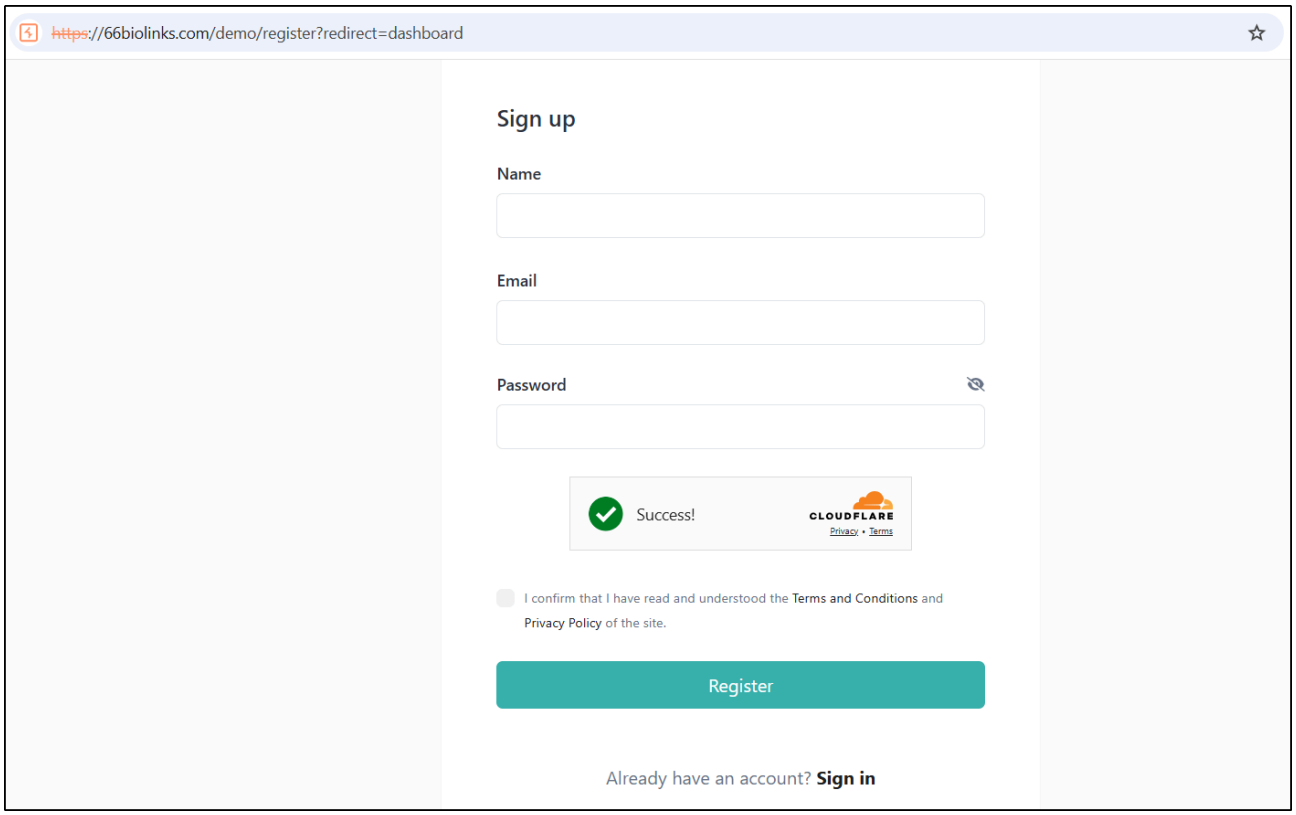
Attacker-supplied SVG favicons allow stored HTML injection, enabling deceptive links or UI elements to appear within users bio links. On visiting favicon URL, external requests occur which exposes users to phishing if they interact with the injected content.

Recommendations

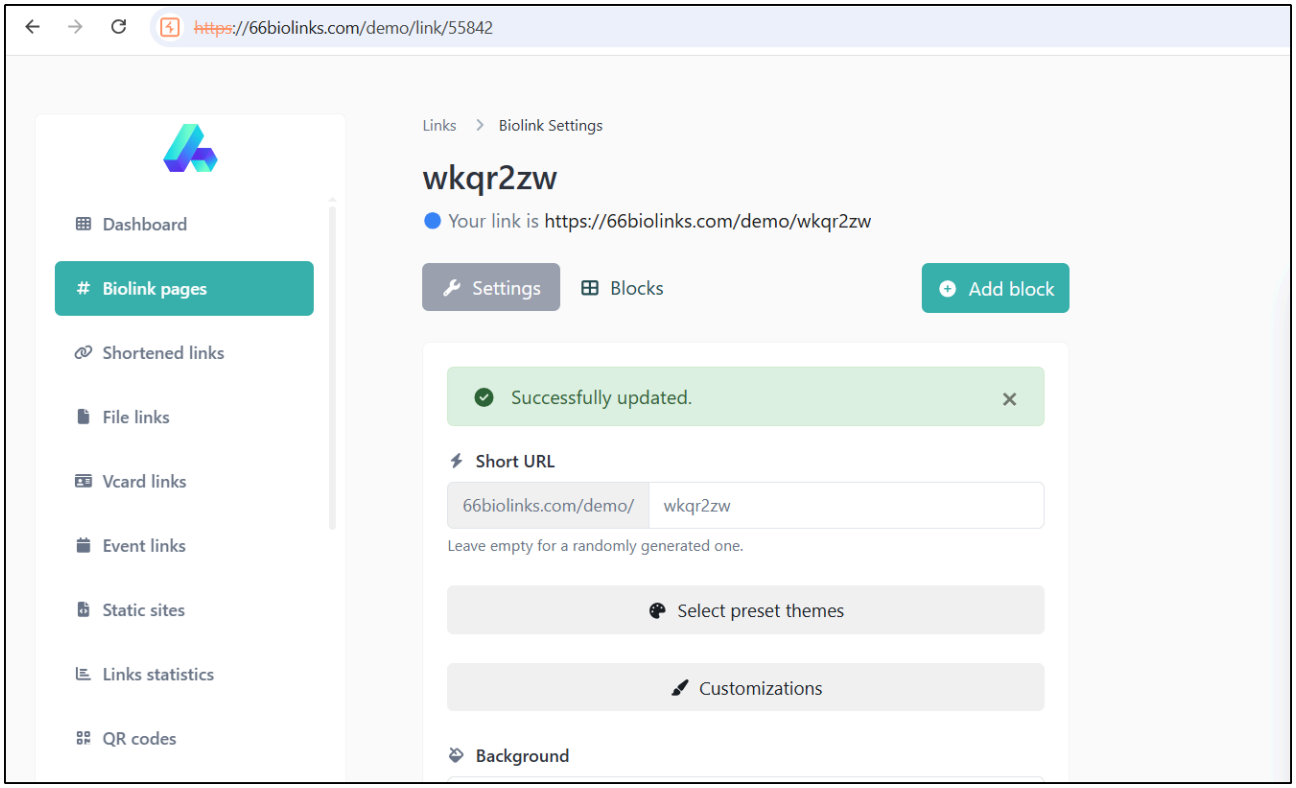
- Strictly sanitize SVG uploads by removing all HTML-capable elements such as and .
- Serve uploaded SVGs as sanitized raster images (PNG/WebP) to eliminate embedded HTML/JS risk.
- Apply output encoding when rendering any user-supplied content inside profile pages.

Proof of Concept (PoC)

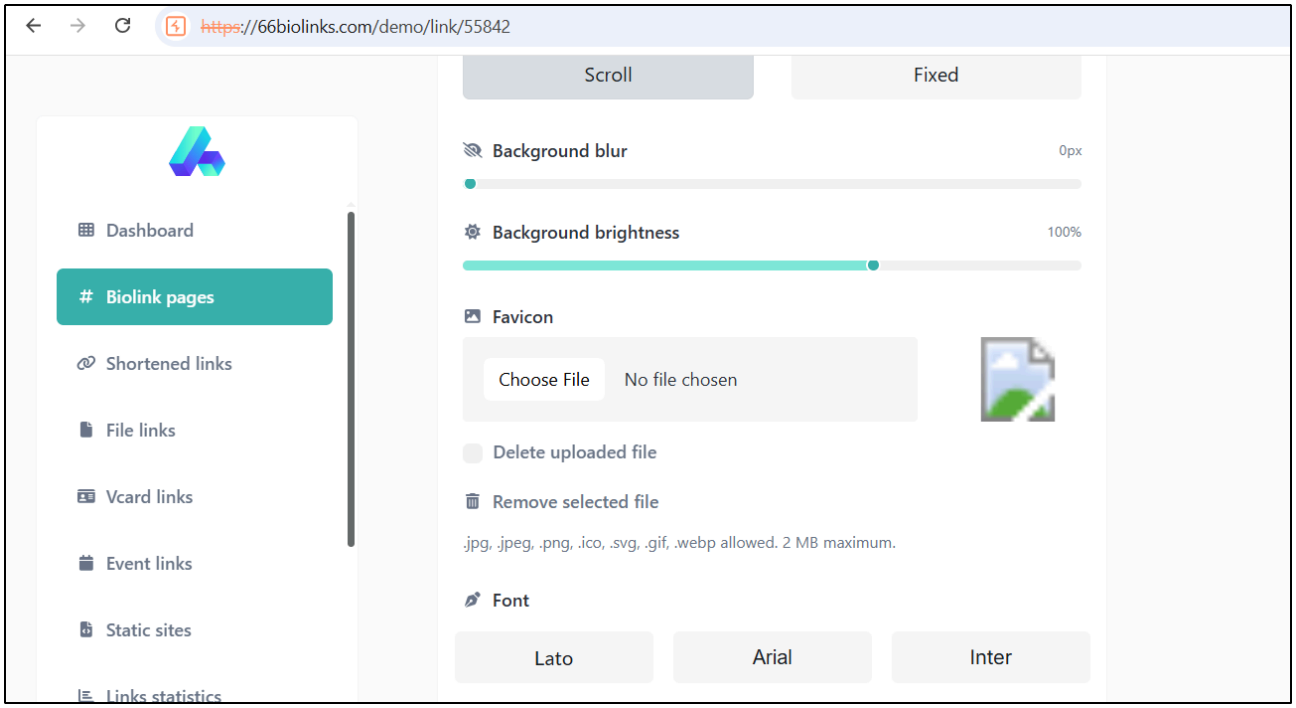
Register a user on 66biolinks portal:



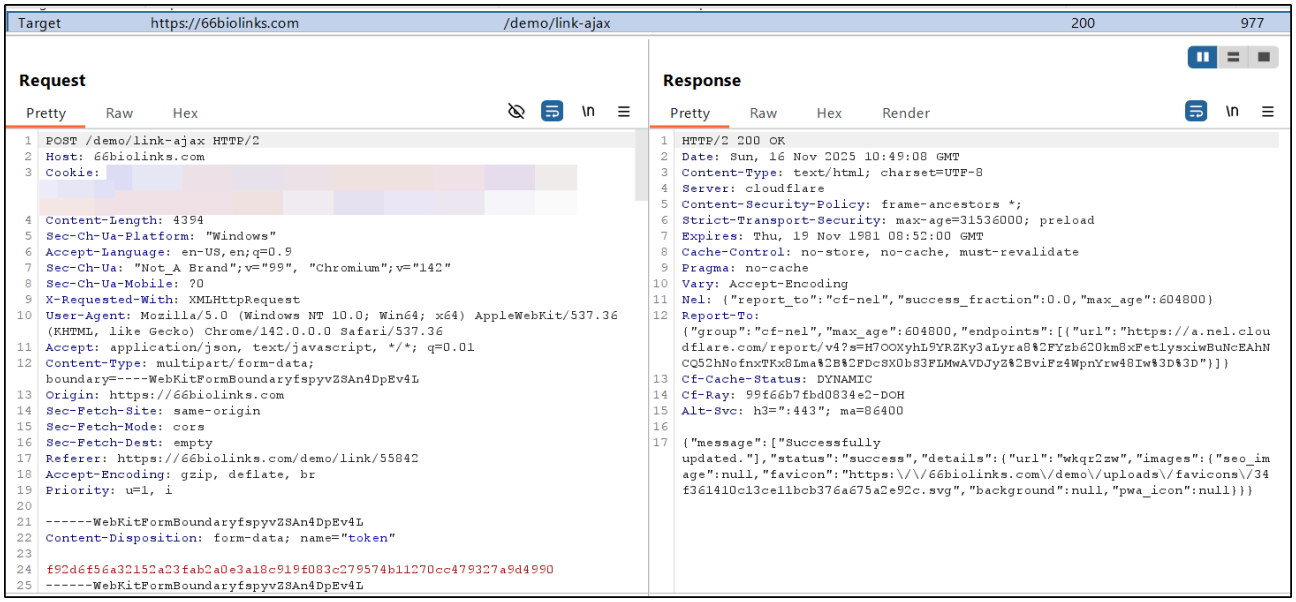
After logging in, navigate to the “Biolink Pages” tab and create a biolink page. After that, click on Settings → Customizations:

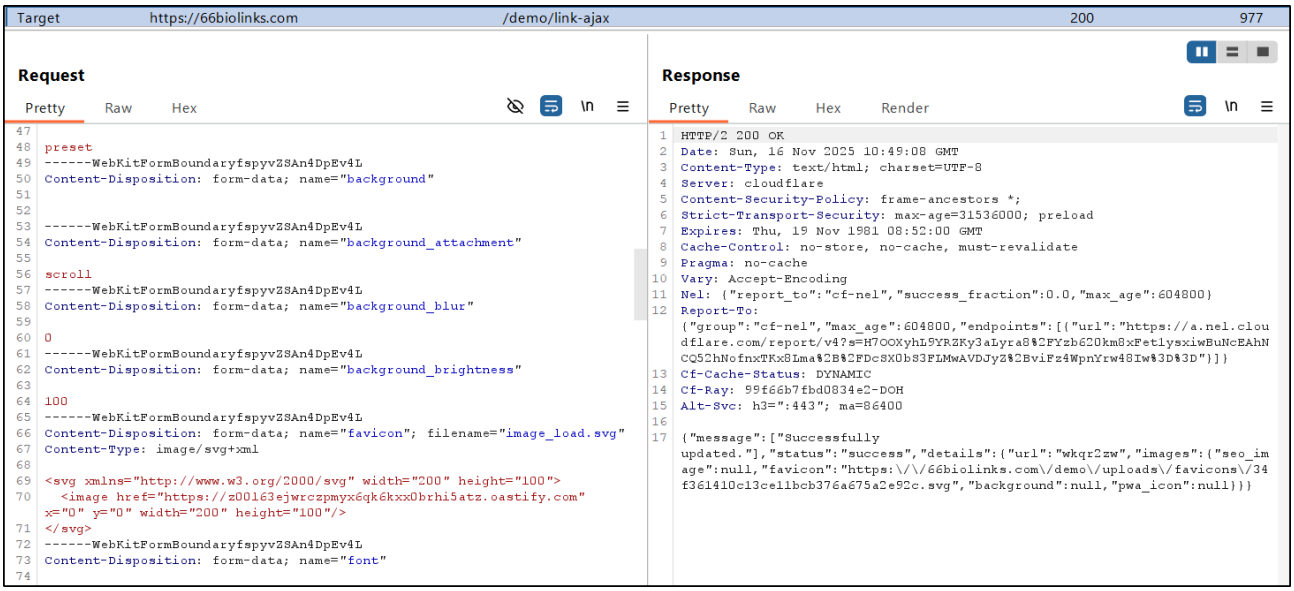


Here, the application allows you to upload an .SVG file as the favicon for the bio page:

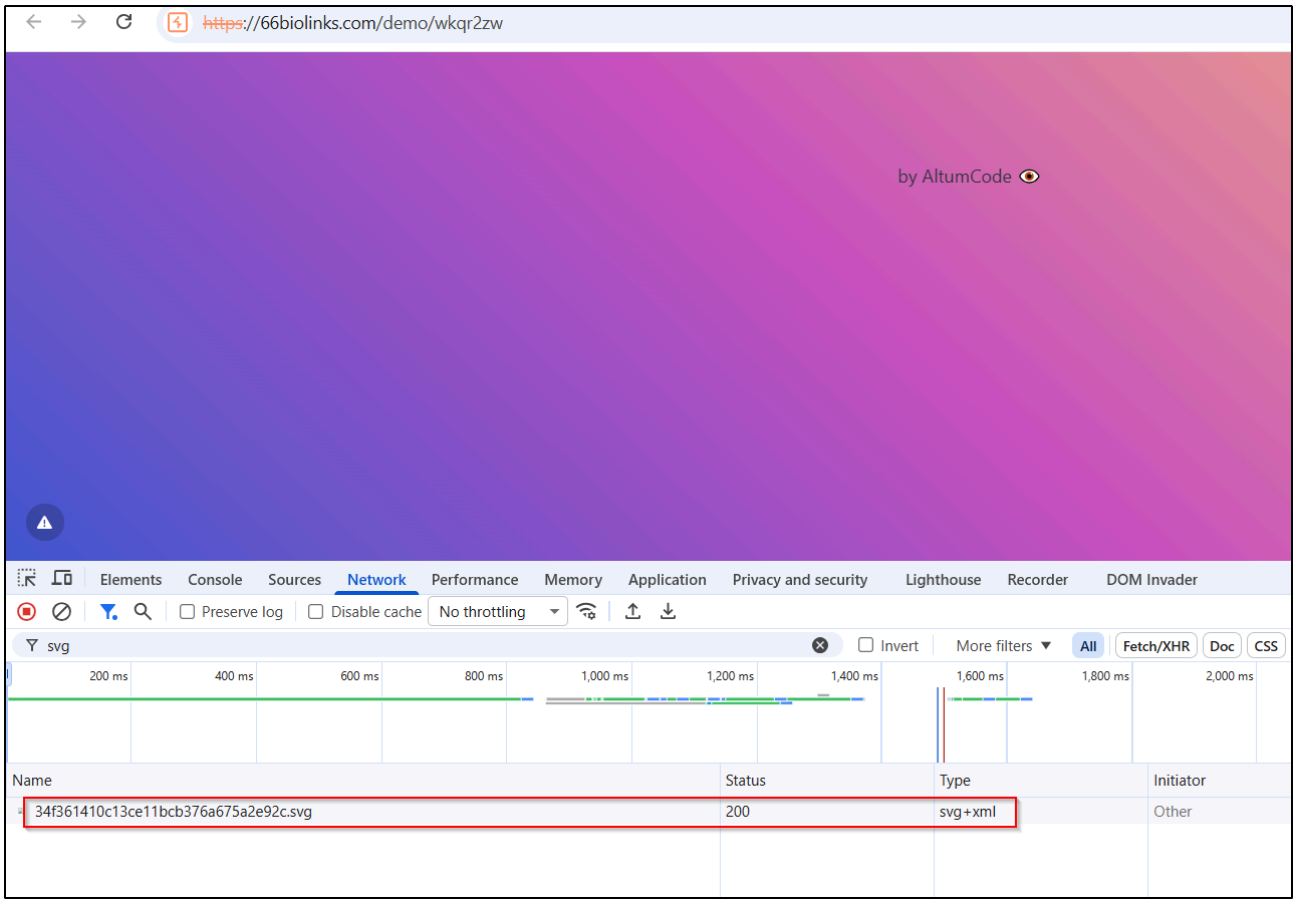


The following request was initiated at the /link-ajax endpoint after submitting changes to the biolink:



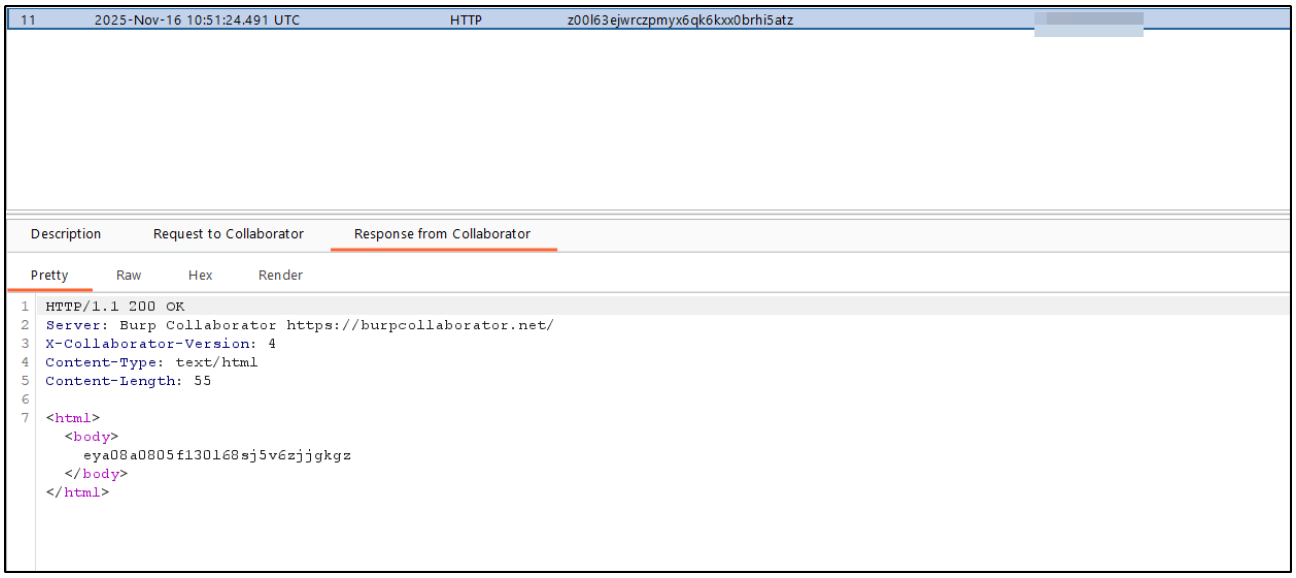


After uploading and saving the biolink page, the link can be visited as shown:



The uploaded payload contained an HTML `<img>` tag, where the reference URL is set to another domain:

After the SVG file is accessed, a request is sent to the specified domain, as shown:

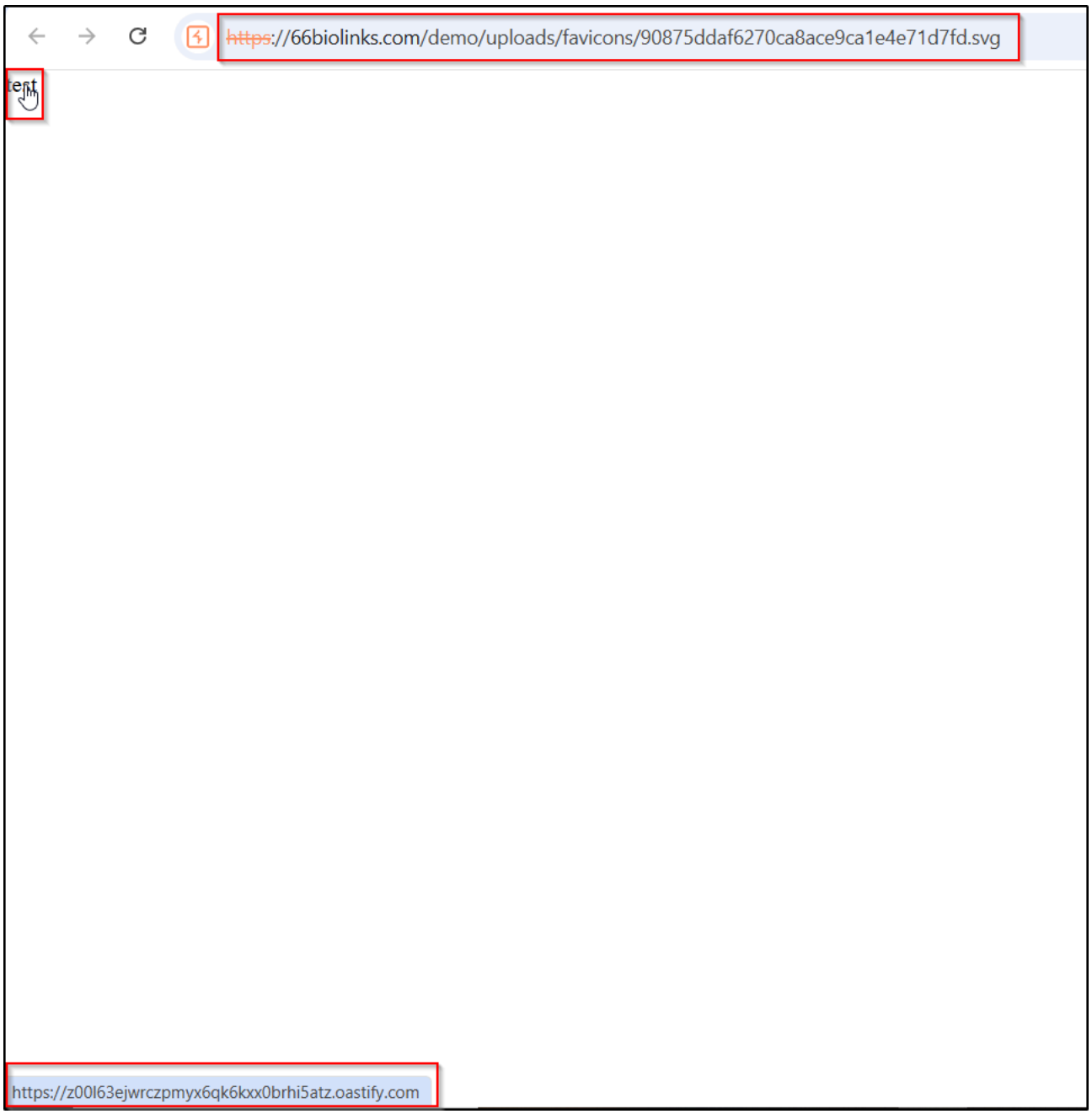


The file upload functionality can also be abused to upload regular images hosted on external domains:


```
<svg xmlns="http://www.w3.org/2000/svg" width="200" height="100">
  <image href="https://example.com/image.png" x="0" y="0" width="200"
```

```
height="100"/>
</svg>
```

Furthermore, HTML anchor tags can be injected into the favicon file, potentially leading to unintended user interactions or navigation:




```
<svg xmlns="http://www.w3.org/2000/svg" width="100" height="20">
  <a href="https://example.com" target="_top">
    <text x="0" y="15">test</text>
  </a>
</svg>
```