Published in System Weakness

Mayur Parmar Follow
Nov 17, 2020 · 2 min read · Listen

CVE-2020–24723

Tale of Stored XSS Leads to admin account takeover

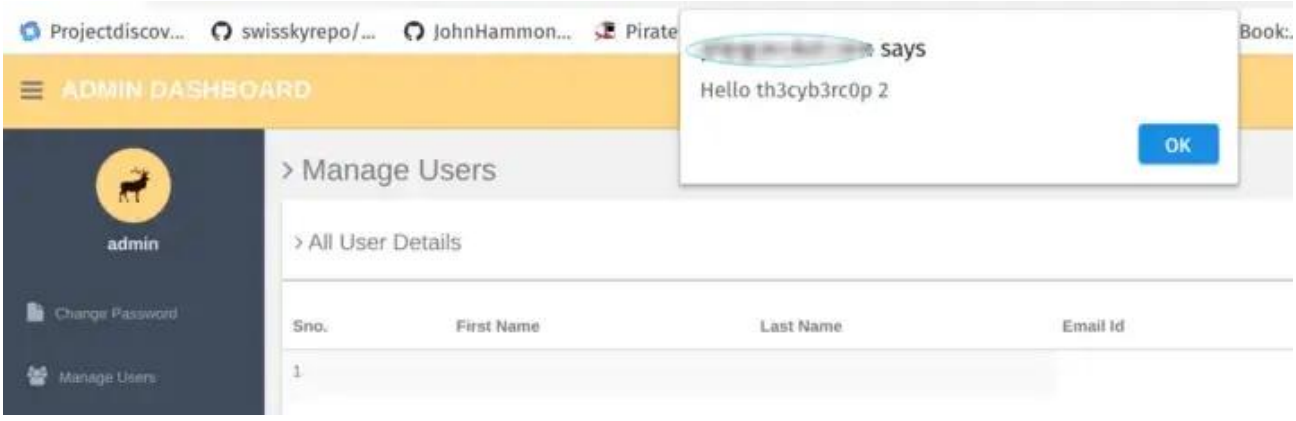
CVE:https://cve.mitre.org/cgi-bin/cvename.cgi?name=2020-24723

 # Exploit Title: User Registration & Login and User Management System 2.1— Stored Cross-Site Scripting
Date: 2020–11–18
Exploit Author: Mayur Parmar(th3cyb3rc0p)
Vendor Homepage: https://phpgurukul.com
Software Link: https://phpgurukul.com/user-registration-login-and-user-management-system-with-admin-panel/
Version: 2.1
Tested on Pop OS(Linux)
 # CVE: CVE-2020–24723

Stored Cross-site scripting(XSS):
Stored attacks are those where the injected script is permanently stored on the target servers,
such as in a database, in a message forum, visitor log, comment field, etc.
The victim then retrieves the malicious script from the server when it requests the stored information.
Stored XSS is also sometimes referred to as Persistent XSS.

Attack vector:
This vulnerability can result in an attacker by injecting the XSS payload in the User Registration section and each time admin visits the manage user section from the admin panel,
the XSS triggers and the attacker can able to steal the cookie according to the crafted payload.

Vulnerable Parameters: Last Name
Steps to reproduce:
1. Goto registration page
2. fill in the details. & put <script>alert(“XSS”)</script> payload in First name,Last name
3. Now go to Admin Panel. we can see that our payload gets executed.



Impact:

any attacker can inject malicious Javascript code and take over the admin account.

Mitigation:

- **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-Content-Type-Options` headers to ensure that browsers interpret [illegible] in the way you intend.

- **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.

> *Author at: https://systemweakness.com/*

Author: Mayur Parmar(th3cyb3rc0p)

https://twitter.com/th3cyb3rc0p?lang=en

https://in.linkedin.com/in/th3cyb3rc0p

https://www.instagram.com/th3cyb3rc0p/?hl=en

https://twitter.com/cyberdefecers?lang=en

Ctf Cve 2020 24723 Bug Bounty Cyber Defecers Ethical Hacking