Menu Management (Posts) Full Account Takeover for All-Roles & Privilege-Escalation via Stored DOM XSS

Critical **bertugfahriozer** published **GHSA-xgh5-w62m-8mpr** 5 days ago

Package

php **ci4-cms-erp/ci4ms** (Composer)

Affected versions	Patched versions
<= 0.28.6.0	0.31.0.0

Description

Summary

Vulnerability: Stored DOM XSS via Posts Added to Menu (Persistent Payload Injection)

- Stored Cross-Site Scripting via Unsafe Rendering of Post Entries in Menu Management

Description

The application fails to properly sanitize user-controlled input when **adding Posts to navigation menus** through the Menu Management functionality. Post-related data selected via the Posts section is stored server-side and rendered without proper output encoding.

These stored values are later rendered unsafely within administrative dashboards and public-facing navigation menus, resulting in stored DOM-based cross-site scripting (XSS).

Affected Functionality

- Menu Management – Posts section
- Adding posts to navigation menus

- Menu storage and rendering logic

Attack Scenario

- An attacker creates or controls a post containing a malicious JavaScript payload.
- The attacker adds the post to the menu using the **Posts** functionality in Menu Manager.
- The application stores the menu entry without sanitization or encoding.
- The payload persists and executes whenever the menu is rendered.

Impact

- Persistent Stored DOM XSS
- Execution of arbitrary JavaScript in victims' browsers
- Privilege escalation in administrative contexts
- Full administrator account takeover
- Full account takeover across all roles
- Full compromise of the entire application via global navigation execution

Endpoint:

- `/backend/menu/`

Steps To Reproduce (POC)

1. Navigate to Menu Management
2. Use the **Posts** section to add a post containing an XSS payload such as:
   `<img src=x onerror=alert(document.domain)>`
3. Save the menu
4. View the menu in the administrative panel or any public-facing page
5. Observe the JavaScript payload executing automatically

Remediation

- Never use .html() again or any innerHTML-style like JS in your PHP, or any other sink, even if user inputs that flow into them are not clear, they still represent real world danger as an attacker can make use of this to exploit the application via XSS. And do HTML Encoding as much as possible and always do Sanitization, theres no sanitization there unfortunately. Also apply CSP, HttpOnly, SameSite, and Secure upon all application, they reduce severity of XSS & escalated-CSRF via XSS and do great jobs

Ready Video POC:

https://mega.nz/file/PcMiUA5K#L2RlZJa340Q8K42TksxiXMuo_9XsRYPi14-WvBnak2A

Severity

Critical 9.1 / 10

CVSS v3 base metrics

Attack vector	Network
Attack complexity	Low
Privileges required	Low
User interaction	None
Scope	Changed
Confidentiality	High
Integrity	Low
Availability	Low

Learn more about base metrics

CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:C/C:H/I:L/A:L

CVE ID

CVE-2026-34565

Weaknesses

▶ CWE-79

Credits

 **bugmithlegend** Reporter

 **peeefour** Reporter