mySites.guru

PageBuilder CK RCE fixed - again - correctly this time

PageBuilder CK's 3.6.0 fix for its file-upload RCE (CVE-2026-56290) only added a login check, so any Editor could still run code up to 3.6.2. Fixed in 3.6.3.

Phil E. Taylor | 21 July 2026

In June 2026 we disclosed a critical flaw in PageBuilder CK, the popular free drag-and-drop page builder for Joomla. It was an unauthenticated file upload that led to remote code execution: anyone, with no login at all, could upload a file to a Joomla site and run it. It became CVE-2026-56290, scored CVSS 10.0, and the vendor shipped version 3.6.0 as the fix.

We went back and read the code the vendor shipped after that. The 3.6.0 fix was incomplete. It added a login check that stops anonymous visitors, and that part works. What it did not do was correct the upload itself. The flaw that let an attacker write and run a PHP file was still there, still in the same file, still with the file-type check commented out, all the way through 3.6.2. All the patch had done was put the bug behind a login. Any account with the Editor permission, which Joomla hands out by default, could still upload and run code. The vendor has now shipped 3.6.3, which restores the missing check and closes it.

This is the difference between patched and fixed, and it is worth understanding, because it is a mistake we see again and again. Update PageBuilder CK to 3.6.3, then read on for what the 3.6.0 fix actually changed and why an “authenticated” upload flaw was still a serious problem.

TL;DR

- The 3.6.0 fix for CVE-2026-56290 only added an **access check**. It stops anonymous visitors, but the **upload handler was not corrected until 3.6.3**
- The **file-extension check stayed commented out** through 3.6.2, so the endpoint would still write a PHP file into a web-served folder
- The remote code execution regressed from **unauthenticated to authenticated**. Any account with **core.edit**, which Joomla grants to the **Editor group by default**, could still reach it

- An Editor is a **low-trust content role** that must never be able to write executable files. So this was a **privilege escalation from editor to full server compromise**
- We confirmed the flaw on a clean test install, uploading a harmless marker file and watching it run, and confirmed that **3.6.3 rejects the same upload**, then cleaned up. **We are not publishing a working exploit**
- **Update to PageBuilder CK 3.6.3**, and as a backstop **block PHP execution in your writable folders** (/images, /media, /templates, /tmp) so an incomplete vendor fix cannot hurt you
- Reported to the vendor, who shipped **3.6.3** with the file-type allow-list restored

How to Find Every Joomla Site Running PageBuilder CK with mySites.guru

When a flaw sits in a component that could be on dozens of your client sites, the first question is always the same: which of my sites run it, and on what version? Logging into every Joomla admin to check the PageBuilder CK version does not scale past a handful of sites.

mySites.guru records the exact version of every installed extension across every connected Joomla site on a twice-daily snapshot. The extension search shows you every site running PageBuilder CK, grouped by version, in seconds. Filter for anything below 3.6.3 and you have your work list.

View all your PageBuilder CK installations

Open PageBuilder CK Extension Search

Lists every installed version across all your connected Joomla sites, so you can find the installations that still need the update.

Combined with the mass extension updater, you can push 3.6.3 across every affected site in one batch. Affected sites are also flagged in the mySites.guru vulnerable

extensions feed, so you are told which sites are exposed rather than having to go looking.

If you do not have a mySites.guru account yet, start a free month and connect your sites. The extension index builds automatically on the first snapshot.

What the 3.6.0 Fix Actually Changed

The original bug had two halves. The endpoint that handles image uploads had no authentication and no permission check, so anyone could reach it. And the upload handler accepted any file, with any extension, and wrote it to a folder the caller chose. Either half on its own is a problem. Together they are unauthenticated remote code execution.

Version 3.6.0 fixed the first half. It added an access check to the component's endpoints, so a request now has to come from a logged-in user with permission to edit before the upload code runs. You can see it working: send an anonymous request to the upload endpoint on a patched site and it comes back with a **403, permission denied**. The anonymous path is genuinely closed, which is why CVE-2026-56290 is correctly marked fixed in 3.6.0.

What 3.6.0 did not touch was the second half. The upload handler did not check the file type. In the code from 3.6.0 through 3.6.2 the extension check was not just missing, it was sitting right there commented out, with a developer's note where the check used to run:

```
// check the file extension // TODO recup preg_match de local dev
// if (getExt($filename) != 'jpg') { ... }
```

The line below it that would reject anything that is not an image was disabled. The handler cleaned the filename but kept the extension, so `shell.php` stayed `shell.php`, and then wrote it into a web-served folder. The file type was looked at only to pick which icon to show in the media browser, never to decide whether the file was allowed.

So the exact flaw that made the June bug a remote code execution was still present, right up to 3.6.2. The 3.6.0 patch built a locked front door and left the back door wide open. Version 3.6.3 finally shuts it, restoring an allow-list that rejects anything that is not an approved media type before the file is written.

A note on what we are not publishing. We confirmed this by reading the shipped code and reproducing the upload on our own test install, with a harmless marker file that we removed straight afterwards, and we verified that 3.6.3 rejects the same upload. No customer or third-party site was involved. We are not naming the exact request or publishing a working exploit. The point is to get people updated and hardened, not to hand a recipe to the next scanner.

Why an “Authenticated” Upload Flaw Is Still Serious

It is tempting to read “now needs a login” as “now safe”. It is not, and the reason is who the login belongs to.

The permission that unlocks the upload is `core.edit`, and in a default Joomla install that permission is granted to the **Editor** user group. An Editor is a content role. It is the account you give to a copywriter, a marketing hire, or a junior team member so they can write and edit articles. It has no business touching the file system, and in a correctly built site it cannot. Joomla’s own media manager deliberately refuses to store executable files for exactly this reason: even a trusted administrator should not be able to drop a PHP file into a web folder by accident, let alone an Editor.

On the vulnerable versions this endpoint handed that power to any Editor. An Editor login, through this one upload, became the ability to write and run arbitrary code on the server. That is a **privilege escalation**: from “can edit an article” to “controls the entire site and everything else it can reach”. Editor accounts are also the softest targets on most sites. They get shared, they get weak passwords, they get reused, and they get phished, precisely because everyone treats them as low-risk. On a site running this code, they were not low-risk at all.

There is a genuine improvement over the June version, and it is worth being honest about it: an anonymous internet-wide attacker can no longer walk straight in. But the gap between "anonymous" and "any Editor" is much smaller than it sounds, and it is nowhere near enough of a barrier for a flaw whose payoff is complete server control.

Being 'patched' is not the same as being 'fixed'

The reason this is worth a whole post, rather than a footnote on the original, is the pattern. A security fix can take two shapes. You can remove the flaw, or you can put a gate in front of it. Removing the flaw here would have meant restoring the file-type check so the handler refuses a PHP upload from anyone. Putting a gate in front of it meant adding a login check so fewer people can reach the unchanged flaw. The 3.6.0 release did the second and not the first, and a critical unauthenticated bug became a serious authenticated one instead of going away.

From the outside these look identical. The changelog says the security issue is fixed, the version number goes up, the CVE is marked resolved. Only reading the code told you the root cause was still there. This is why, when we track a vulnerability, we care about what the fix actually did and not just that a fix shipped.

It is also the strongest possible argument for **defence in depth**. If your writable directories cannot execute PHP in the first place, then a handler that wrongly writes a `.php` file has written an inert text file, and an incomplete vendor fix never becomes your incident. Block PHP execution in `/images`, `/media`, `/templates` and `/tmp` at the web server level, not in the application, so the protection holds regardless of what any one extension does. If you run Akeeba Admin Tools, its `.htaccess` Maker sets up most of this for you. It is the layer that would have blunted both the June bug and this one - but even this is still just locking the fridge and the toilet and letting the hacker sit on your sofa.

What To Do

Update PageBuilder CK to 3.6.3 on every Joomla site that runs it. In each site's Joomla admin, go to **System** then **Update** then **Extensions**, find PageBuilder CK, and update. If the update does not appear there, download the latest build from joomlack.fr and install it over the top. If you run more than a handful of sites, push the update from your mySites.guru dashboard rather than working through admin panels one by one.

Then harden the site so you are not relying on any single extension getting its file handling right. Block PHP execution in your writable directories, review your **Users** list and remove any Editor or Super User accounts you do not recognise, and tighten the passwords on the ones you keep. If a site has been running this code with untrusted Editor accounts, treat a compromise as possible and run a full security audit: look for stray PHP files under `/images`, `/media`, `/templates` and `/administrator`, and if you find one, clean the site properly and rotate your Joomla passwords and secrets.

The Same Root Cause, One Layer Down

The June bug fit a pattern we have written about at length: front-end endpoints that check a token but never check permission. This one is a variation on it. The permission check finally got added, but the dangerous operation behind it, an upload with no file-type control, was left untouched, so the flaw simply changed who could reach it.

It sits alongside the other file-upload flaws we have covered this year, several of which are the same shape. Phoca Download was an authenticated upload where a member-level account could bypass the allow-list and run code, a close cousin of this Editor-level flaw. Balbooa Forms, iCagenda and the SP Page Builder zero day were unauthenticated versions of the same idea. We pulled the whole pattern apart in AJAX endpoints are a big CMS security blind spot, and it keeps proving relevant.

The lesson is not "stop using extensions". PageBuilder CK is genuinely good at its job, and this is a fixable oversight, not a reason to rip it out. The lesson is that the security of your sites rests on code other people wrote and ship on their own schedule, that even a shipped fix is worth checking, and that the day a flaw becomes public you need to know, within minutes, which of your sites run it. That is the whole reason mySites.guru indexes every extension on every site you connect. The corrected release

is here in 3.6.3. You want to be the operator who updated and swept before the next scanner comes round, not the one finding a strange file three weeks later.

CVE Record

The original unauthenticated flaw is tracked as CVE-2026-56290, published 29 June 2026, CVSS 10.0, CWE-284, crediting Phil Taylor of mySites.guru as the finder. The residual authenticated flaw described here is an incomplete fix of that same issue. We reported it to the vendor on 21 July 2026, and the vendor released 3.6.3 with the file-type allow-list restored, which we confirmed on a clean test install. No separate CVE has been assigned for the authenticated variant; this section will be updated if one is.

Field	Detail
Original CVE	CVE-2026-56290 (unauthenticated, fixed in 3.6.0)
This finding	Incomplete fix of CVE-2026-56290: upload handler wrote executable files through 3.6.2
Reach	Authenticated. Any account with `core.edit` (Editor group by default)
Component	PageBuilder CK (`com_pagebuilderck`)
Vendor	Cedric Keiflin (joomlack.fr)
Type	Authenticated arbitrary file upload to remote code execution (privilege escalation)
Still present in	3.6.2 and earlier of the current line
Fixed in	3.6.3 (restores the file-type allow-list)
Reported	21 July 2026 (fixed in 3.6.3)

Further Reading

- The original PageBuilder CK unauthenticated upload RCE (CVE-2026-56290) - the June flaw this one grew out of.
- AJAX endpoints are a big CMS security blind spot - the wider pattern of endpoints that check a token but not permission.
- A month of Joomla security disclosures - the full run of extension flaws in one place.
- PageBuilder CK by Cedric Keiflin - the vendor product page.
- OWASP: Unrestricted File Upload - the vulnerability class, and how to prevent it properly.
- Joomla security best practices - including blocking PHP execution in writable folders.

Frequently Asked Questions

Is PageBuilder CK still vulnerable after the 3.6.0 fix?

It was, up to and including 3.6.2, and it is fixed in 3.6.3. The 3.6.0 release fixed CVE-2026-56290 by adding an access check that stops anonymous visitors reaching the upload endpoint, and that part works: a request with no login gets a 403. But the upload code itself was not corrected. The file-extension check stayed commented out, so the endpoint would still write a PHP file into a web-served folder. The 3.6.0 patch moved the bug behind a login rather than removing it. Version 3.6.3 restores the missing check, so update to it.

Who can exploit the residual PageBuilder CK flaw?

Any account with the core.edit permission on the component. In a default Joomla install that permission is granted to the Editor group. An Editor is a low-trust content role, the sort of account you hand to a copywriter or a junior team member so they can edit articles. It is never supposed to be able to write executable files to the server. Because this endpoint let an Editor upload and run a PHP file, an Editor login became full remote code execution: a privilege escalation from content editor to complete server control. Version 3.6.3 closes it.

The endpoint needs a login and a CSRF token. Doesn't that make it safe?

No. Requiring a login raised the bar from 'anyone on the internet' to 'anyone with an Editor account', but Editor is far too low a bar for remote code execution. Editor accounts are common, shared, and often handed out freely, and they are the exact accounts most likely to be phished or reused with a weak password. An anti-CSRF token stops the action being triggered from another site, but it does nothing to stop the logged-in Editor doing it deliberately. The right control, an allow-list that refuses executable file types, is the one 3.6.3 finally restores.

How do I fix it?

Update PageBuilder CK to 3.6.3 on every Joomla site that runs it. You can update through the Joomla admin under System then Update then Extensions, push the update across many sites at once from your mySites.guru dashboard, or download the latest build from joomlack.fr and install it over the top. As defence in depth that survives an incomplete vendor fix, also block PHP execution in your writable directories (/images, /media, /templates, /tmp) at the web server, so a file dropped there cannot be run even if something writes it.

What is the difference between this and CVE-2026-56290?

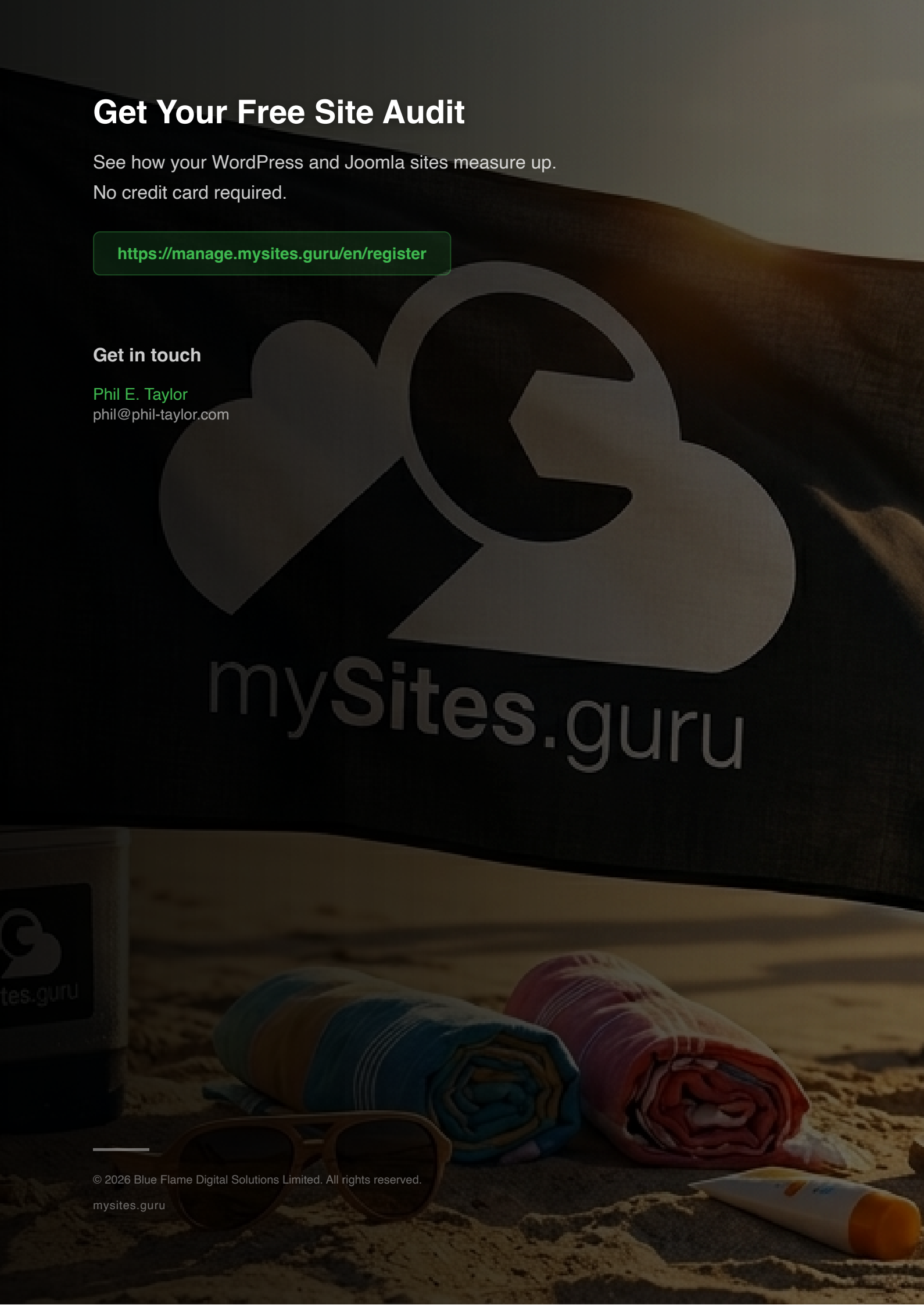
Same bug, same file, different reach. CVE-2026-56290 was the unauthenticated version: anyone could reach the upload with no login at all. The 3.6.0 fix added the login and permission check that closed the anonymous path, which is why the CVE is marked fixed in 3.6.0. What the fix did not do was correct the upload handler, which still accepted any file type and wrote it to a web-served folder. So the same remote code execution stayed reachable, now by an authenticated Editor rather than an anonymous visitor, right up to 3.6.2. It is best understood as an incomplete fix of CVE-2026-56290, resolved in 3.6.3.

Get Your Free Site Audit

See how your WordPress and Joomla sites measure up.

No credit card required.

https://manage.mysites.guru/en/register



Get in touch

Phil E. Taylor
phil@phil-taylor.com