mySites.guru

Balbooa Forms Fixes an Unauthenticated File Upload RCE

Balbooa Forms up to 2.4.0 (com_baforms) had an unauthenticated file upload flaw that let anyone drop a PHP file and run code. CVE-2026-56291, fixed in 2.4.1.

Phil E. Taylor | 9 July 2026

Balbooa Forms is a popular drag-and-drop form builder for Joomla, installed as the `com_baforms` component and used for contact, registration, and survey forms on thousands of sites. Up to and including version 2.4.0, its frontend attachment upload had a serious flaw: it accepted a file from any anonymous visitor, with no login, no CSRF token, and no check on the file type. An attacker could upload a `.php` file into a public folder and then run it, which is unauthenticated remote code execution, the worst outcome a web flaw can have. This was a zero-day: it was already being exploited in the wild when we found it, before any patch existed, and those attacks are still going on now against sites that have not updated.

Balbooa fixed it in version 2.4.1, released on 9 July 2026. If you run Joomla sites with Balbooa Forms on them, update every one to 2.4.1 or later right now. This is not a wait-for-a-maintenance-window update. Any site still on 2.4.0 or earlier is being actively attacked, so treat it as urgent and check it for tampering the moment it is patched. This post explains what the flaw was, how it surfaced, what the fix does, and how to check whether any of your sites were touched, without publishing anything an attacker could copy.

> **Actively exploited right now, update immediately**
>
> This was a zero-day. Attackers were exploiting it in the wild before Balbooa had a patch, and those attacks are still running against any site left on 2.4.0 or earlier. Update every Balbooa Forms install to 2.4.1 without waiting for a maintenance window, then check each patched site for stray files and unexpected admin accounts, because a site could already have been hit.

TL;DR

TL;DR: The frontend upload in Balbooa Forms (`com_baforms`) up to and including 2.4.0 (fixed in 2.4.1) accepted a file from anyone with no authentication, no CSRF token, and

no allow-list on the file extension. Because it trusted the caller's filename, a `.php` upload landed in a public directory and could be executed, which is unauthenticated remote code execution (CWE-434). This is a zero-day that was, and still is, being actively exploited in the wild: it came to light through a Hetzner abuse report that a mySites.guru customer brought to us as a raw access log, evidence of a live attack, and there was no patch at that point. We diagnosed it, disclosed it privately to Balbooa, and they responded the same day and shipped the fix in 2.4.1 within a day, crediting Phil Taylor as the reporter. It is tracked as CVE-2026-56291. No proof of concept has been made public. Update every Balbooa Forms install to 2.4.1 or later immediately, then check exposed sites for stray files and unexpected admin accounts. If you manage more than a handful of Joomla sites, mySites.guru lists every Balbooa Forms install in your account in seconds and pushes the update.

How This One Was Found

Most vulnerability posts start with a CVE and work backwards. This one started with a log line.

A customer received an abuse report from Hetzner, the hosting company, about suspicious activity on one of their Joomla sites. Rather than guess at it or panic, they did the right thing: they brought us the raw web server access log and asked us to look. Credit to Hetzner for catching the activity and flagging it, and to the customer for bringing evidence instead of a hunch. A single log line is worth more than a page of speculation.

That log showed a `POST` request to the Balbooa Forms upload handler that had no business succeeding the way it did. We traced the request back through the extension's code, confirmed that the frontend upload accepted files with no authentication and no restriction on type, and reproduced the whole chain end to end on a local Joomla install within the hour. From there it was a private email to Balbooa, who replied the same day and shipped the fix in version 2.4.1 the next day, on 9 July 2026, crediting Phil Taylor as the reporter. The flaw is now tracked as CVE-2026-56291.

This is what responsible disclosure looks like

Found through a real abuse report, confirmed in the code, reported privately, and fixed by the vendor within a day, all before any technical detail was published. No exploit has been released. The reason to write about it is to get sites updated, not to show anyone how it worked.

What Was Actually Wrong in Balbooa Forms?

The flaw was an unauthenticated file upload with no file-type allow-list. The Balbooa Forms upload endpoint runs for any visitor, logged in or not, and before the fix it did three things wrong at once: it required no login, it checked no CSRF token, and it worked out the file extension from the name the visitor supplied and used it as-is. Joomla's built-in filename cleaner strips dangerous characters, but it does not block a `.php` extension, so a file named `shell.php` stayed `shell.php`.

Put those together and an anonymous request could write an attacker-controlled `.php` file into the extension's upload folder, which by default sits under `images/baforms/uploads/` inside the web root. Because that folder is served directly and has nothing stopping PHP from running in it, requesting the uploaded file in a browser executed it. Upload becomes code execution, and no account on the site is needed at any point.

The important detail for site owners is the "no login" part. Plenty of upload bugs need an editor or admin account first, which limits the blast radius to people you already trust. This one did not. Anyone on the internet who could reach the form could reach the upload handler.

Root Cause, In the Code

For the developers among you, here is the mechanism without the exploit. The request enters through a single front-end task, `index.php?option=com_baforms&task=form.uploadAttachmentFile`. In 2.4.0 the controller that handles it ran no authentication check and no `Session::checkToken()` call, so the

task was reachable by any anonymous visitor and accepted a raw file straight from the request.

The controller passed that file to the model's write routine (`FormModel::uploadAttachmentFile`, around line 122 in the front-end `FormModel.php`). That routine is where the real problem sat. It worked the file extension out from the attacker-supplied filename, sanitised only the name portion with Joomla's `File::makeSafe()`, and then re-attached the original extension verbatim before writing the file. In effect:

1. the extension is taken from the caller's filename, so `shell.php` yields `php`;
2. `File::makeSafe()` cleans the name but does not reject a `.php` extension;
3. the cleaned name and the untouched extension are joined back together (`$fileName = $name . '.' . $ext`);
4. the file is written under `images/baforms/uploads/form-<id>/`, a directory inside the web root that serves and executes PHP.

There was no allow-list checking the extension against the file types the form is actually meant to accept. That single missing check is the whole vulnerability. With no login, no token, and no allow-list standing in the way, an anonymous upload of a `.php` file became code running on the server. We are deliberately not publishing the request body that ties these together, but the shape of the bug is the same one this class of extension keeps producing.

Why an Upload Flaw Is as Bad as It Gets

An unauthenticated upload that lands executable code is the top of the severity scale, and it is worth being clear about why. Once an attacker can run PHP of their choosing on your server, they are no longer limited to your site. They can read your `configuration.php` and the database credentials in it, create a hidden Joomla Super User, plant a backdoor that survives a plugin update, pivot to other sites in the same hosting account, or add the server to a spam or malware network. The uploaded file is rarely the goal in itself; it is the foothold.

This is the same lesson as the AJAX endpoints blind spot we keep coming back to. An endpoint that Joomla will run for anonymous visitors is exposed to the entire internet, so every input it touches has to be treated as hostile. A single allow-list check, "is this file one of the types this form is allowed to accept", would have stopped this outright.

What the 2.4.1 Fix Does

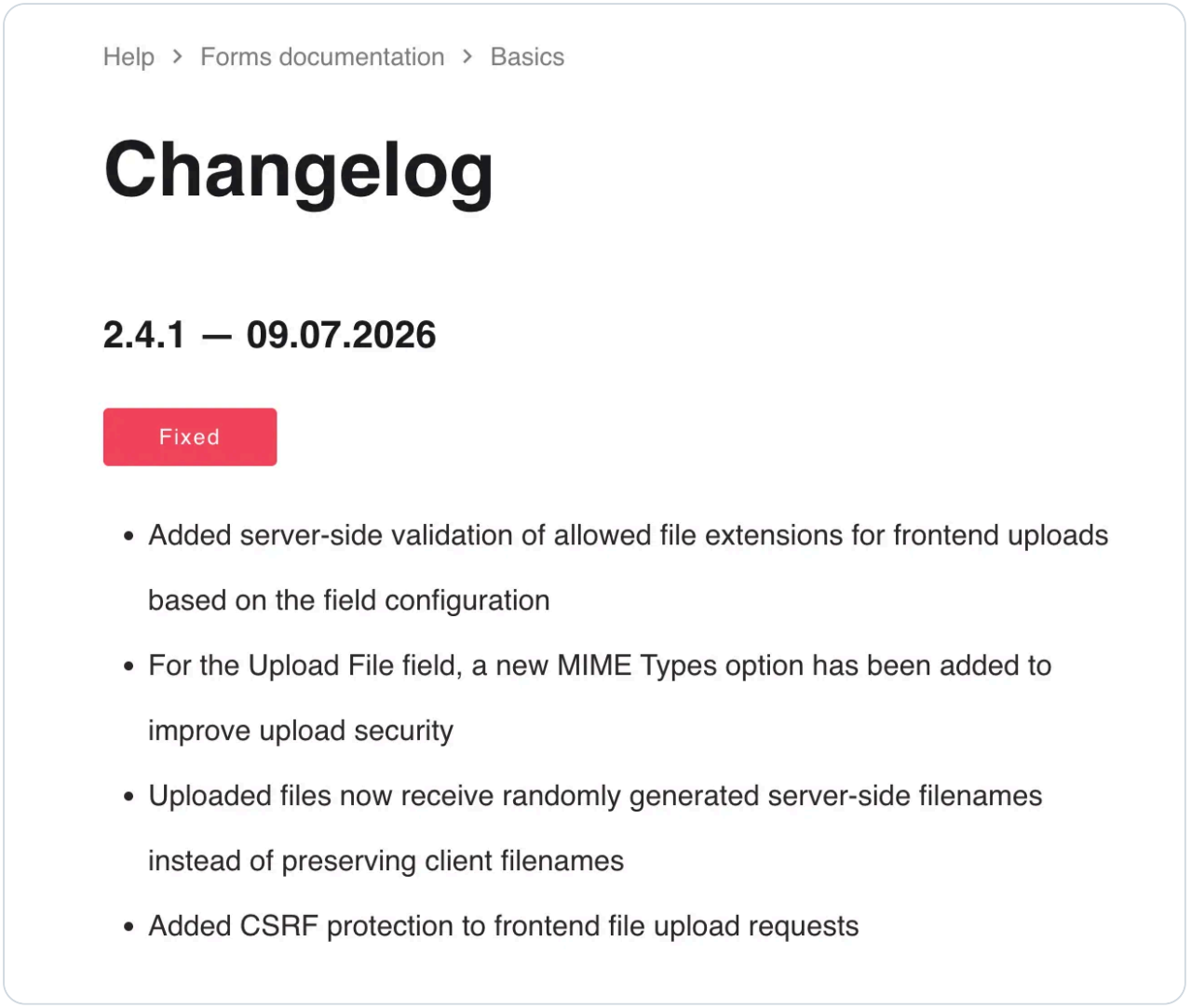
Balbooa's fix is a good one, and it closes the flaw at more than one layer, which is exactly what you want. Version 2.4.1 makes four changes to the frontend upload:

- It now validates the file extension on the server against each Upload File field's own configuration, so a `.php` upload to a field that only accepts images is refused.
- It adds a MIME Types option to the Upload File field, so a form can also require the file's real content type to match, not just its name.
- It generates the stored filename on the server instead of trusting the one the visitor sent. Even a file that gets through as an allowed type is saved under a name the attacker cannot predict, which defeats the classic double-extension trick.
- It adds a CSRF token check to the upload request, so the handler no longer runs for a request that did not come from a real form on the site.

We verified all four against the released package. On 2.4.0, an anonymous request with no token uploaded a `.php` file and ran it. On 2.4.1 the same request is rejected outright, and even a request carrying a valid token cannot land executable code, because the extension allow-list and the forced filename both stand in the way. The fix holds.

There is one thing worth flagging for anyone who reads vendor changelogs to decide what to update. Balbooa's 2.4.1 changelog lists these four items under a plain "Fixed" heading, with no mention that they close an actively exploited remote code execution flaw. That is not unusual, and it is not a criticism, but it is a trap. If you triage updates by whether the changelog says "security", this one reads like routine housekeeping, and you would leave it sitting in the queue while the door stays open. It is the same pattern

we saw with the Helix3 “Security Update” changelog: the words in the changelog are a poor guide to how urgent an update really is.



Worth adding: Balbooa did update the changelog, but at the time of writing there is no post on the Balbooa Forms blog about this security issue at all. So the only public signal that 2.4.1 matters is four terse lines under a “Fixed” heading, with nothing anywhere that tells a site owner this closes an actively exploited remote code execution flaw. If you are waiting for a vendor announcement before you update, there is not one to wait for.

Balbooa Forms Has Had Security Issues Before

This is not the first time Balbooa Forms has needed a security fix, which is worth knowing when you decide how urgently to act. The component has two prior SQL injection issues on record, CVE-2021-47930 and CVE-2025-49485, both reached through the form submission `id` parameter. Different flaw, same lesson: the parts of a form builder that face anonymous visitors are the parts that need the hardest scrutiny.

That is not a knock on Balbooa specifically. It is the nature of the tool. A form builder exists to accept input from people you do not know, and that job is genuinely hard to do safely. The credit here is that when the upload flaw was reported, Balbooa acted on the same day rather than sitting on it.

Form Builders Are a Recurring Attack Surface

If you manage Joomla or WordPress sites, treat every form-builder extension as a piece of your attack surface that deserves regular attention. The exact flaw fixed here, an unauthenticated file upload of a dangerous type, is CWE-434, and it shows up across the form-builder category rather than in one vendor's code.

The clearest recent parallel is Convert Forms, another well-regarded Joomla form builder, which patched an unauthenticated file upload of the same class in late 2024 (CVE-2024-40744, rated 9.8 critical). Look wider and the pattern holds: Ninja Forms on WordPress had its own file-upload CVE through the same AJAX-handler weakness. When a category of extension keeps producing the same kind of bug, the sensible response is not to distrust one vendor, it is to monitor the whole category.

How mySites.guru Caught This Without a Signature

If you already run sites through mySites.guru, this is the part worth knowing. We did not need a Balbooa-specific rule to catch this activity. mySites.guru watches for the behaviour, an anonymous visitor uploading an executable file, rather than a fingerprint of one particular extension's bug.

That generic detection is what flagged the suspicious upload in the first place, and it is the same logic that catches iCagenda, Page Builder CK, and JCE upload attempts. An attacker dropping a `.php` file through a public form looks the same regardless of which extension left the door open, so we can catch new bugs of this shape before anyone has written a rule naming them. Signature-based tools only see an attack once someone has described it. Behaviour-based detection sees it the first time it happens.

If a hostile file does land, the suspect content tool and the backdoor and hacked-file detection find it across every connected site, matching it against known malware hashes and thousands of code patterns. Anything flagged can be sent for AI-powered malware analysis that explains in plain English what the file does.

How Do I Find Every Balbooa Forms Site I Manage?

The first question after any extension security release is the awkward one: which of my sites actually run this? Up to about ten sites, you can log in to each Joomla admin and check the installed extensions list. Past that, you need a single view.

mySites.guru keeps a live inventory of every extension, template, and framework on every Joomla and WordPress site in your account. You search for Balbooa Forms once and get back every connected site running it, the version each one is on, and whether an update is available. No logging into forty admin panels one at a time.

View every Balbooa Forms install across your sites

Open your Extension Inventory

Search for Balbooa Forms across every connected Joomla site and filter for anything on 2.4.0 or earlier to find the installs that still need updating. Not a subscriber? Sign up free and connect your sites.

Across the Joomla sites we monitor, the same thing is true of Balbooa Forms as of most extensions after a quiet security release: a lot of installs are running older versions and nobody has updated yet. That gap between “a fix exists” and “my sites have it” is the whole reason this post exists.

Bulk Updating Balbooa Forms Across Every Site

Once you know which sites need it, the mass updater handles the rollout. Tick the sites on an old version, push the update to all of them from one screen. The same routine covers any Joomla extension, plugin, or core update, so the workflow you set up once works for the next security release too. You can also switch on automatic updates for any Joomla extension so future Balbooa Forms releases land without you lifting a finger.

> **For agencies managing dozens of Joomla sites**
>
> The patch is the easy bit. Knowing which client sites run Balbooa Forms, and getting the update onto all of them, is the work. See how mySites.guru manages multiple Joomla sites from one screen.

If a particular site cannot be updated right away, unpublishing any public Balbooa Forms form that accepts file attachments takes the vulnerable upload handler out of reach while you schedule a maintenance window.

How Do I Check a Balbooa Forms Site for Tampering?

Updating closes the door. It does not tell you whether anyone walked through it first. Because the flaw allowed an anonymous visitor to drop a file and run it, a site that was on 2.4.0 or earlier could already have been touched, and you should check before you assume it is clean.

Three checks, in order of value:

1. Look in the upload folder. By default Balbooa Forms stores attachments under `images/baforms/uploads/`, in a subfolder per form. Anything there that is not an image or a document, and above all anything ending in `.php`, is a red flag. On a single site you can run `find images/baforms/uploads -name '*.php' -type f` over SSH.

2. Check for rogue administrator accounts. Code execution is a fast route to a hidden Super User. In your Joomla Users list, sort by registration date and treat any administrator account you do not recognise as suspect.
3. Hunt for modified and unfamiliar files. A foothold is a perfect moment to plant persistence elsewhere. Look for recently changed PHP files and for executable files sitting where uploads should not contain code.

mySites.guru runs all three of these across every connected site at once. The suspect content tool and hacked-file detection surface the stray PHP files, the extension and user inventory sorts every account across every site by registration date, and real-time alerting tells you the moment a new file appears or an unfamiliar admin logs in, rather than at the next manual check.

If any of that turns something up, the Joomla hacked recovery guide and the how to fix a hacked site walkthrough cover the cleanup, and fix.mysites.guru is the done-for-you option if you would rather hand it over.

Stay Ahead of the Next One

This is one extension on one day. There will be another, because Joomla runs on thousands of third-party extensions and the ones that accept input from anonymous visitors keep producing bugs like this. The hard part is never the update itself. It is knowing a fix exists, knowing which of your sites are affected, and getting to them before an attacker does, and doing that for every extension across every site you look after.

That is the job mySites.guru does for you. It keeps a live inventory of every extension on every Joomla and WordPress site in your account, flags the ones with a known vulnerability, and lets you push the update to all of them from one screen. When something like this Balbooa Forms flaw lands, you see exactly which sites are exposed in seconds instead of logging into forty admin panels to find out. And because the monitoring watches for the behaviour rather than a signature, it catches new upload attacks of this shape before anyone has written a rule naming them.

Get free email alerts when a Joomla vulnerability breaks

We email a plain-English alert the moment a serious flaw like this one is disclosed, with the affected versions and what to do. No charge, unsubscribe any time.

Subscribe to security alerts

Want the alerts and the tooling to act on them? Start with a free audit on one site and see your full extension inventory, or sign up for mySites.guru to get vulnerability alerts and one-click updates across every site you manage.

CVE Record and Disclosure Timeline

10.0
CVSS 4.0

CRITICAL **The maximum possible score**

There is no worse rating a vulnerability can get. It is unauthenticated, remotely exploitable in a single request, needs no user interaction, and ends in full remote code execution, so every metric that makes a flaw dangerous is at its worst here.

No login needed | Exploitable over the internet | No user interaction | Full remote code execution | Actively exploited

This flaw is now tracked as CVE-2026-56291, assigned by the Joomla CNA, the body that assigns identifiers for Joomla and its extensions, and crediting Phil Taylor of mySites.guru as the reporter. It is CWE-434, unrestricted upload of a file with a dangerous type, reached by an anonymous visitor over the network in a single request, with no privileges and no user interaction, and it ends in full remote code execution. On CVSS 4.0 that profile scores 10.0, the maximum, the same score and the same vector as the near-identical Page Builder CK flaw (CVE-2026-56290), the CVE assigned one number earlier for the same class of bug.

Field	Detail
CVE	CVE-2026-56291

Field	Detail
Component	Balbooa Forms (`com_baforms`)
Vendor	Balbooa
Type	Unauthenticated arbitrary file upload to remote code execution
CVSS 4.0	10.0 (critical), `AV:N/AC:L/AT:N/PR:N/UI:N/VC:H/VI:H/VA:H/SC:H/SI:H/SA:H`
CWE	CWE-434 (Unrestricted Upload of File with Dangerous Type)
Finder	Phil Taylor, mySites.guru
Affected versions	Up to and including 2.4.0
Fixed in	2.4.1, released 9 July 2026

The disclosure ran on a single-day cycle from evidence to fix:

Date	Event
8 July 2026	A mySites.guru customer receives a Hetzner abuse report for a live Joomla site and brings us the raw access log. We trace the `POST` to the Balbooa Forms upload handler, confirm the missing authentication and file-type checks in the code, and reproduce the full chain on a local Joomla install. The flaw is already being exploited in the wild. We disclose it privately to Balbooa the same day, and the vendor responds the same day.
9 July 2026	Balbooa ships version 2.4.1, closing the flaw at four layers (server-side extension allow-list, MIME type option, forced server-side filenames, and a CSRF token check), crediting Phil Taylor as the reporter. The Joomla CNA assigns CVE-2026-56291. No proof of concept is published.

Further Reading

- CWE-434: Unrestricted Upload of File with Dangerous Type - the canonical definition of this weakness class from MITRE.
- OWASP File Upload Cheat Sheet - the developer's checklist for accepting uploads safely: allow-lists, signature checks, renaming, and storing outside the web root.

- OWASP: Unrestricted File Upload - a concise overview of the risk and the defences.
- PortSwigger Web Security Academy: File upload vulnerabilities - a neutral, in-depth reference on how these flaws work and how to prevent them.
- Securing Joomla extensions - Joomla's own guidance for developers, including safe file handling with `JFilterInput` and `JFile::makeSafe()`.

Frequently Asked Questions

What was the security flaw in Balbooa Forms?

The frontend attachment upload in the Balbooa Forms component (com_baforms) accepted a file from anyone, with no login, no CSRF token, and no check on the file type. Because the extension trusted the filename the visitor supplied, an anonymous attacker could upload a .php file into a public folder and then request it in a browser to run their own code on the server. That is unauthenticated remote code execution, the most serious outcome a web flaw can have. It is classed as CWE-434, unrestricted upload of a file with a dangerous type.

Which version of Balbooa Forms is affected, and which one fixes it?

The flaw was present in Balbooa Forms up to and including 2.4.0. Balbooa fixed it in 2.4.1, released on 9 July 2026. Update every Balbooa Forms install to 2.4.1 or later. If you are on 2.4.0 or earlier, treat the site as exposed until it is updated, and check it for tampering because this flaw was being exploited in the wild before the fix shipped.

Is this a zero-day?

In effect, yes. The flaw was already being exploited in the wild when it was found, through a real abuse report on a live site, and no fix existed at that point. What we did not do was publish it. The vendor was told privately, responded the same day, and shipped 2.4.1 within a day, all before any technical detail was made public. This post still withholds the exact request an attacker would send. The point of publishing is to get sites updated, not to hand anyone an exploit.

How do I find every Balbooa Forms install across the sites I manage?

Manually you would log in to each Joomla site and check the installed extensions list. With mySites.guru you open the extension inventory, search for Balbooa Forms, and get every connected site running it with its installed version on one screen. Anything on 2.4.0 or earlier needs the update, and you can push it to all of them from the same place.

My site runs Balbooa Forms. How do I know if it was already hit?

Updating stops the next attempt but does not undo one that already happened. Look in the Balbooa Forms upload folder (by default images/baforms/uploads) for any file that is not an image or document, especially anything ending in .php. Check your Joomla user list for administrator accounts you do not recognise, and look for recently modified or unfamiliar

PHP files across the site. mySites.guru automates all three checks across every connected site.

Do form-builder extensions get attacked often?

Yes. A form builder is designed to accept input from anonymous visitors, including file uploads, which makes it a natural target. The same class of flaw, an unauthenticated file upload, was fixed in Convert Forms in late 2024 (CVE-2024-40744, also CWE-434). Any extension that takes uploads from the public needs a strict allow-list of file types, and most of the serious Joomla incidents we help clean up start at exactly this kind of endpoint.

Does mySites.guru need a signature for each new exploit like this?

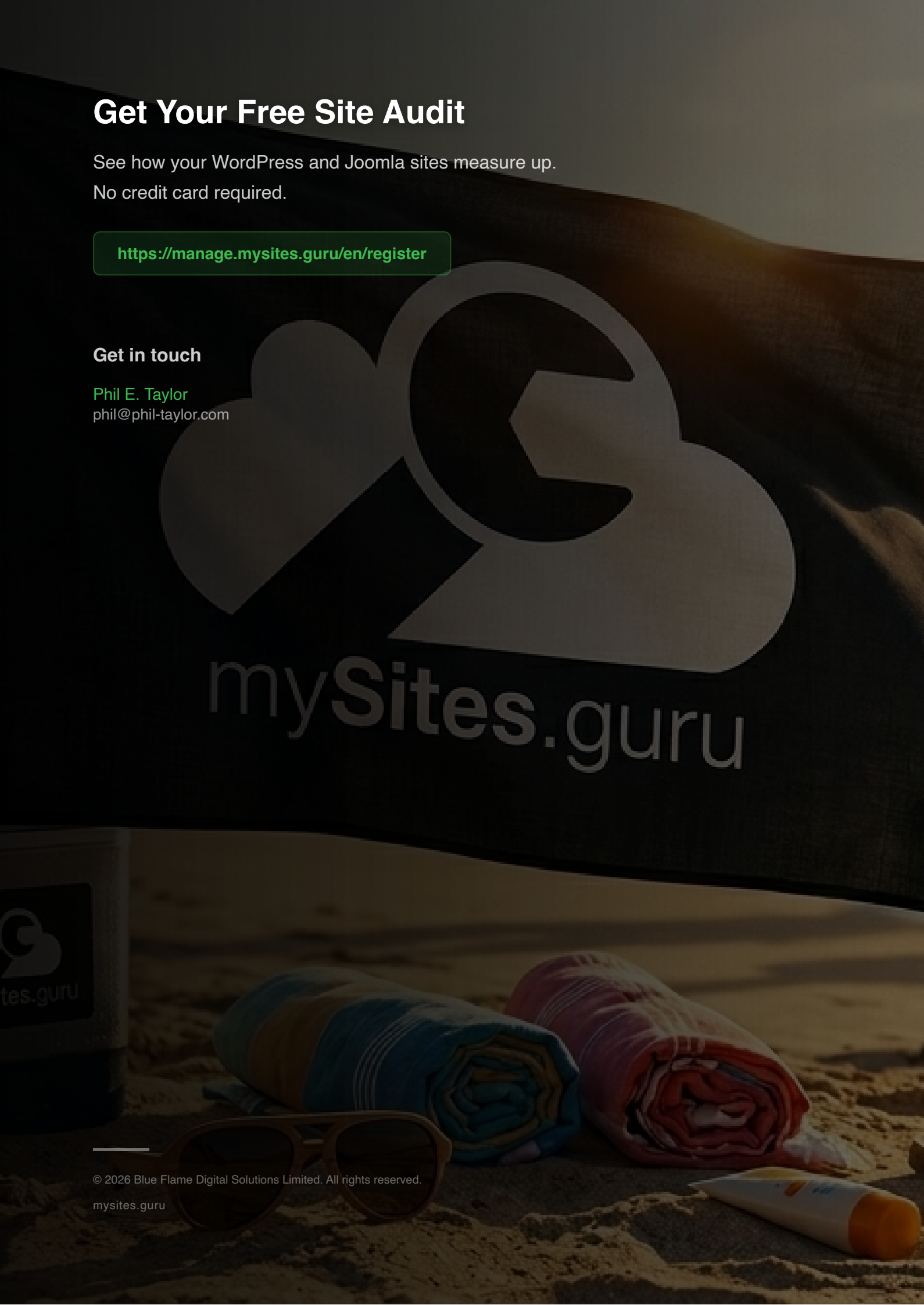
No, and that is the point. mySites.guru watches for the behaviour, an anonymous visitor uploading an executable file, rather than a fingerprint of one specific extension's bug. That generic detection caught this Balbooa Forms activity the same way it flags iCagenda, Page Builder CK, and JCE upload attempts, without anyone writing a Balbooa-specific rule first.

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